AD-A092 441 AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH F/G 14/2
DIAGNOSTIC SOFTWARE DEVELOPMENT FOR A REAL-TIME SPECTRAL ANALYS--ETC(U)
APR 80 B J DONLAN
UNCLASSIFIED AFIT-CI-80-9T NL

1 OF 2
AD
A092441

UNCLASS

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER
1--80-9T

2. GOVT ACCESSION NO.
AD-A092 441

3. RECIPIENT'S CATALOG NUMBER

4. TITLE (and Subtitle) Diagnostic Software Development For A Real-Time Spectral Analysis System

(10)

5. TYPE OF REPORT & PERIOD COVERED
THESIS/~~DISSERTATION~~

6. PERFORMING ORG. REPORT NUMBER

7. AUTHOR(s)
Brian J. Donlan

8. CONTRACT OR GRANT NUMBER(s)

9. PERFORMING ORGANIZATION NAME AND ADDRESS
AFIT STUDENT AT: Rensselaer Polytechnic Institute

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS

11. CONTROLLING OFFICE NAME AND ADDRESS
AFIT/NR
WPAFB OH 45433

(11)

12. REPORT DATE
Apr~~il~~ 1980

13. NUMBER OF PAGES
105

14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)

15. SECURITY CLASS. (of this report)
UNCLASS

15a. DECLASSIFICATION DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES
APPROVED FOR PUBLIC RELEASE: IAW AFR 190-17

Fredric C. Lynch
FREDRIC C. LYNCH, Major, USAF
Director of Public Affairs

Air Force Institute of Technology (ATC)
Wright-Patterson AFB, OH 45433

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)
ATTACHED

ABSTRACT

This report deals with the development of diagnostic software for a real-time spectral analysis system. The report gives a description of the spectral analysis system and its associated maintainability and reliability problems. The diagnostic system is comprised of a read-only memory board containing a small operating system and diagnostic tests. Detailed descriptions and listings of the tests and operating system are provided.

DIAGNOSTIC SOFTWARE DEVELOPMENT FOR A REAL-TIME SPECTRAL ANALYSIS SYSTEM

by

Brian J. Donlan

A Project Submitted to the Graduate
Faculty of Rensselaer Polytechnic Institute
in Partial Fulfillment of the Requirements
for the Degree of
MASTER OF SCIENCE

Approved by:

G. Robert Redinbo, Advisor

Rensselaer Polytechnic Institute
Troy, New York

April 1980

AFIT RESEARCH ASSESSMENT

The purpose of this questionnaire is to ascertain the value and/or contribution of research accomplished by students or faculty of the Air Force Institute of Technology (ATC). It would be greatly appreciated if you would complete the following questionnaire and return it to:

AFIT/NR
Wright-Patterson AFB OH 45433

Research Title: Diagnostic Software Development for a Real-Time Spectral Analysis System

Author: Brian J. Donlan

Research Assessment Questions:

1. Did this research contribute to a current Air Force project?

a. Yes b. No

2. Do you believe this research topic is significant enough that it would have been researched (or contracted) by your organization or another agency if AFIT had not?

a. Yes b. No

3. The benefits of AFIT research can often be expressed by the equivalent value that your agency achieved/received by virtue of AFIT performing the research. Can you estimate what this research would have cost if it had been accomplished under contract or if it had been done in-house in terms of manpower and/or dollars?

a. Man-years ______. b. $__________

4. Often it is not possible to attach equivalent dollar values to research, although the results of the research may, in fact, be important. Whether or not you were able to establish an equivalent value for this research (3 above), what is your estimate of its significance?

2. Highly Significant b. Significant c. Slightly Significant d. Of No Significance

5. AFIT welcomes any further comments you may have on the above questions, or any additional details concerning the current application, future potential, or other value of this research. Please use the back of this questionnaire for your statement(s).

NAME GRADE POSITION

ORGANIZATION LOCATION

CONTENTS

LIST OF TABLES

LIST OF FIGURES

Page

ACKNOWLEDGEMENT

The author wishes to express his gratitude to Dr. G. Robert Redinbo and Dr. Lester A. Gerhardt for their advice and guidance.

Special thanks to the Air Force Institute of Technology for their academic sponsorship.

Finally to Corey and Joey for taking many naps and allowing their mother to type the manuscript.

Section 1

Introduction-History

Over the past few years the researchers here at Rensselear have been designing and constructing a Real-Time Spectral Analysis System. This system was designed to be used by oceanographers in their studies of the tides and tidal erosion. The primary purpose of the system is to measure the very low frequency power spectra of the ocean waves. The power spectral density is obtained by taking the time varying input signals from the oceanographer's sensors and transforming it from the time domain to the frequency domain using a Fast Fourier Transform. The resulting spectral information is presented on a color graphics display.

In the summer of 1978 the newly completed system was taken to a beach in Florida for a field test. The test proved partially unsuccessful. The basic system ideas and functions were correct, but the system proved difficult to test and keep operational. Reliability and maintainability became the system's downfall.

In the haste to complete the system for the summer tests very little was done to facilitate testing of fault diagnosis of the system or individual components. The author accepted the task of remedying the situation by developing some diagnostic programs and methods which will aid in test and diagnosis of the Real-Time Spectral Analysis System.

This report presents the author's efforts and the diagnostic programs developed to solve the given problem.

Since no tests or test procedures existed, the entire diagnostic system was the author's responsibility.

Section 2

System Description

2.1 General Description-Data flow

The real-time spectral analysis system is composed of 4 major subsystems as presented in figure 2.1.

Analog data from up to four sensors is input to the data acquistion subsystem where it is amplified, filtered and digitized. Next, the digitized input data is transfered to the high speed array processor where a Discr- te Fourier Transform is performed on the data string converting it from the time domain to the frequency domain.

The frequency domain spectral data is transfered to the color graphics display where it is displayed in a time verses frequency format with color encoding representing amplitude

Detailed descriptions of each subsystem and its functions follow.

2.2 IMSAI Microcomputer and Front Panel

The IMSAI Microcomputer containing an Intel 8080 CPU is the heart of the system. The IMSAI thru the Unibus adapter controls many of the systems and options. The 8080 operates the user front panel, receiving and displaying the many parameters and options, and passes them out to the required subsystem. The microcomputer also functions as a host for the Floating Point Array Processor, which has the task of loading the signal processing programs into the array processor.

In its present configuration, the IMSAI system contains 24K bytes of semiconductor random access memory, an 8" floppy disk drive and interface, a serial input/output port for console communication, and two parallel input/output boards used in the Unibus adapter. The IMSAI also hosts a small numerical processor. With this project, an additional 16K byte read-only memory board was added to the IMSAI to contain the diagnostic tests. The 8080 also contains an adapter board which enables it to simulate a DEC PDP-11 computer and perform data transfers over the Unibus.

2.3 Floating Point AP-1208 Array Processor

The Floating Point Array Processor performs the actual signal processing. The digitized input data is transfered to the array processor where it is first multiplied by a user selected window. The windowing helps minimize any distortion caused by the time limiting of the input data. The data is then transformed into the frequency domain using a 1024 point Fast Fourier Transform (FFT). After the Fast Fourier Transform (FFT) the spectral data may be filtered or further processed before being presented to the color graphics display.

The AP-120B is a very fast and versitale floating point array processor with a basic cycle time of 167 nanoseconds. The AP-120B has a pipeline structure with a 38-bit floating point format. The instruction words are 64 bits long so many functions can be performed in one machine cycle. This type of speed and instructions format is ideal for the reiterative additions and multiplication often used in digital signal processing. For example, a 512 point FFT can be performed in 3.2 MSEC.

The array processor is a slave computer and requires a host to operate it. In this system the IMSAI 8080 functions as the Host. The AP has no front panel and all access to the AP is thru a Unibus interface.

2.4 Data Acquisition

The present system has four analog input channels. The amplified analog inputs are filtered by a programmable anti-aliasing filter to band limit the input high frequency components. The input signal high frequencies must be limited to prevent aliasing. The filter cutoff frequency is set by the IMSAI 8080 depending on the sampling frequency selected on the user front panel. The analog signal is then sampled by an A/D converter which converts the continous input data to sampled digital data. The system is capable of sampling frequencies of from .001 HZ to 15 KHZ. The input circuitry contains a programmable real-time clock used to control the sampling rate. This clock is set by the IMSAI in response to the front panel selected frequency. The digitized data is transfered directly to the array processor data memory via an AP input/output port interface.

2.5 Color Graphics Display

The color graphics display system was built especially for this system. The graphics display portion consists of a 512 x 512 point screen format which is stored in a large Intel refresh memory.

The frequency spectrum of each input block of 1024 input samples is displayed on one horizontal line with frequency increasing from left to right. A time history of the spectrum is shown in the vertical direction as each new line is added. As each FFT computation is completed in the array processor its spectrum is output into one horizontal line on the display. The new line is added to the bottom of the screen and the older lines are scrolled up presenting the history of the spectrum. Each point of the line is color amplitude encoded with one of 128 possible colors.

Two lines of characters for anotation purposes are provided at the top of the screen. Under the two lines of anotation are two color bars used in the color amplitude encoding. The top most bar is called the menu and it presents all 128 possible colors. The lower bar is the color map which contains 64 locations into which a color can be loaded. Each map position represents an increment of magnitude of the frequency spectrum. The color placed in each map location representing a certain level of magnitude is selected on the user front panel. The entire map may be stored on the floppy disk for quick reloading.

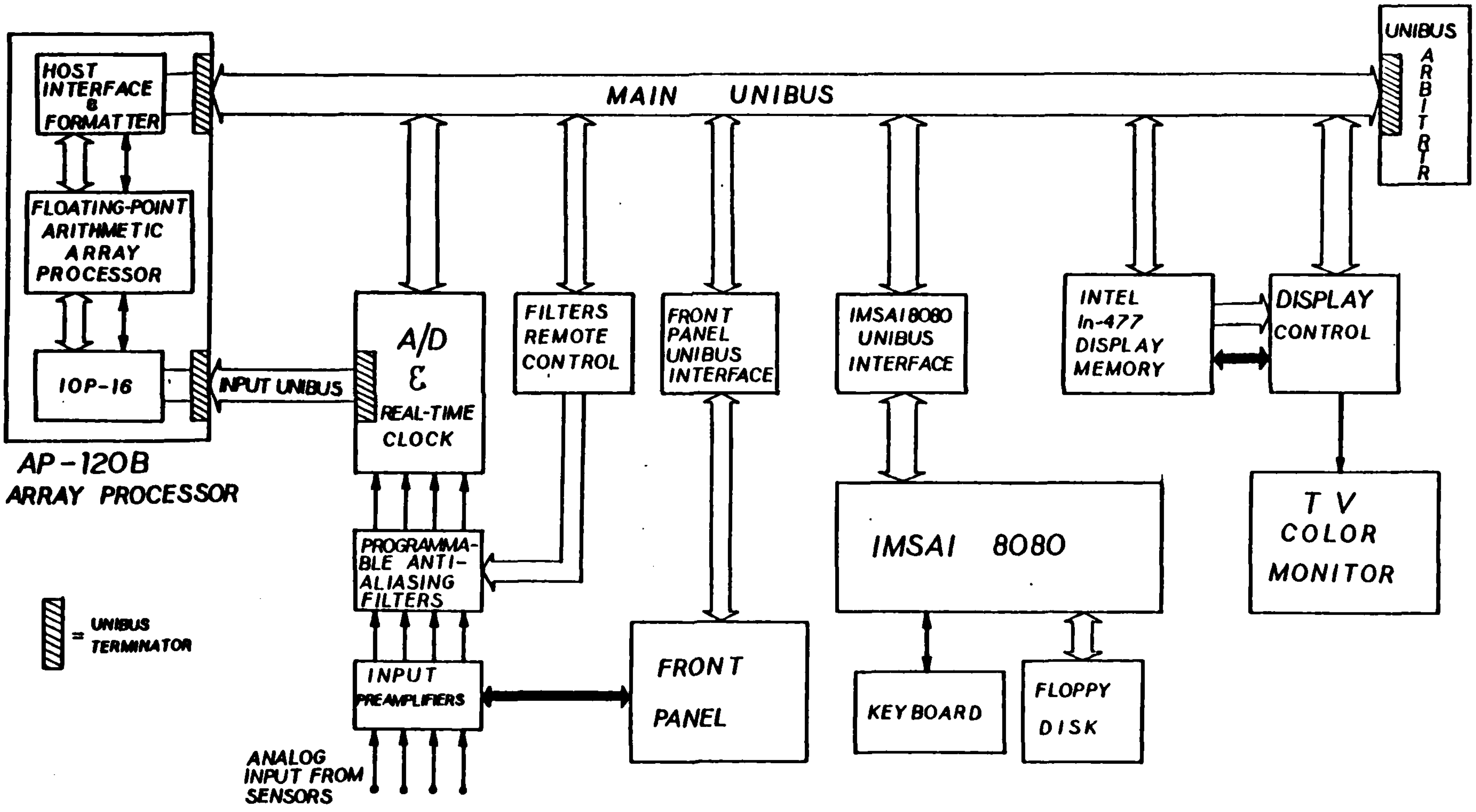


FIG 2.1 SYSTEM BLOCK DIAGRAM

Section 3
Technical Discussion

3.1 Task

The purpose of this project is to identify potential system malfunctions and to design software and hardware aids which will help in isolating system faults. History has helped in locating many problem areas. The spectral analysis system has a number of weak hardware links which are prone to failure. These areas received attention first.

3.2 Design Details

The floppy disk system has been a consistent problem area. The disk drive is a very delicate device which was never mount or placed in a case, leaving it exposed to contamination and physical abuse.

In the original system the disk was the keystone of the spectrum analysis system. All memory was volatile and once powered-down left the system void of intelligence. The only mass storage device from which to load any programs or diagnostics was the disk drive, making a disk failure catastrophic. A programmable read-only memory containing a small operating systems and diagnostics, was added to the IMSAI 8080 computer. This enables the computer system to always have a small operating system available, even in the event of a disk failure.

The floppy disk is probaby one of the most difficult parts of the system to troubleshoot. The dik controller requires many driver routines to read or write to the disk drive. A large disk diagnostic was written and stored on the diagnostic memory board to aid in troubleshooting and testing the disk system. Two smaller programs were also written to aid in disk drive alignment and testing.

The IMSAI 8080 computer is another area where a possible failure leads to complete system failure. The IMSAI uses an Intel 8080 microprocessor as CPU. Because of the one chip simplicity and cheap replacement, diagnostics on the actual CPU instuctions did not seem prudent. Diagnostics to test the computer memory were written however. The computer system presently has 24K of MOS RAM memory. The MOS memory is very sensitive to static charges and power supply voltage variations. A comprehensive memory test using an advanced test algorithm by Knaizut and Hartmann [2] was implemented. A simpler mini-memory test has the advantage of not requiring a console device or scratch pad memory. These tests are all resident on the diagnostic memory board.

The majority of the devices in the spectral analysis system are handwired prototype devices connected to the Pseudo-Unibus. The IMSAI microcomputer has control of the Unibus via a Unibus to S-100 bus adapter. The adapter board uses 6 I/O ports in the IMSAI 8080. Since problems with shorted Unibus lines have been common in the past, a diagnostic was written specifically to test the I/O ports and Unibus Lines.

In order to communicate with the many devices on the Unibus, a Unibus communication test was written. This is probably the most useful of all tests written; the user need only input the device address and a transfer command. A data word can be transfered to or from any device on the Unibus. Timers are also included in the program to test for device 'no answers', a common problem with Unibus devices.

The last major component of the system is the Floating-Point array processor. At present, a complete diagnostic package for the AP exists. This package was delivered with the AP and runs on the DEC PDP-11. The spectral analysis system also has a small AP debug program as part of the real-time program. No further diagnostics were written for the AP. The Unibus communication test can be used to read and write to the array processor front panel.

3.3 Test Procedures

Since the heart of the system is the IMSAI 8080 computer, no part of the system will function without it, making it the first area to be treated.

The following is a recommended test sequence:

1. Mini-Memory test 0 to 100 hex version — this tests the scratch pad memory used by other tests and the diagnostic operating system. This test needs no console. (sec.5)

2. Mini-Memory Test 24K version — this provides a quick test of all memory. Don't forget to check the power supplies if you have problems. (sec.5).

3. Diagnostic Operating System — try some simple command for overall CPU operation. This also tests the console for overall CPU operation. This also tests the console I/O operation. (sec.4)

4. Comprehensive Memory Test — this provides a good test on the memory and should catch most problems. (sec.6)

5. CPM disk operating system — boot the disk system and see what happens. This provides a good indication of overall disk operation. Most disk problems show up here.

6. Formatted disk test — this test tests the normal formatted operation of the disk. This test can take a long time to check all 77 tracks and a bad diskette media can cause a failure; use a good blank disk.

7. Unibus Port Test — This checks for shorted unibus lines and broken wires. If this fails look for an incorrectly inserted card in the lower card cage. (sec.10)

8. Unibus Communication Test — try to communicate with the various unibus devices. (sec.11)

9. Real-time Spectral Analysis program — the ultimate test.

Section 4

Diagnostic Operating System

4.1 General Description

In order to enable the computer system to have some capabilities and intelligence during a major system failure, a small self-contained operating system is included on the diagnostic memory board. This operating system can be used to run the diagnostic program as well as perform a number of standalone function such as memory and register examine.

4.2 Detailed Description

The diagnostic operating system is a modified and expanded version of the SSM 8080 Monitor V-1 supplied with the Prom Memory board. The operating system supplied was modified to handle our I/O requirements and the scratch pad area used by the operating system was fixed to the first 256 bytes of RAM memory. A diagnostic test directory and controller were added to ease diagnostic program execution. A Help command was also added to remind the casual user of the various commands and options available.

4.3 Program Usage

The starting address of the operating system is F000 Hex and it can be started directly from the IMSAI 8080 front panel. The operating system is located in a readonly memory on the diagnostic memory board and requires only a minimum system of 256 byte of ram memory and a console device to function.

The operator communication with the monitor consist of a single alphabetic character input on the console device and may be followed by one or more parameters. When two or more parameters are used they are separated by a coma or a space. Parameters are hexadecimal values consisting of four or two hexadecimal characters. Leading zeros are assumed. The command line is terminated by carriage return in most cases.

4.4 Commands

The following is a modified and expanded explanation of the SSM monitor commands

D Command- (Display memory)

D'Low Address','High Addrees'

Memory from 'Low Address' through 'High Address' is displayed on the console device. If 'High Address' is equal to or smaller than 'Low Address', only the 'Low Address' byte is displayed. Data bytes are displayed in hexadecimal, 16 bytes per line. The beginning address of each line is displayed.

S Command - (Subsitute memory)

S'Address'

The byte at location 'Address' is displayed on the console device followed by a - character. The operator responds with one or more characters from the console. If the input character is a space or comma, the contents of the next location is displayed. If one or more hexadecimal digits are inputed before the space or comma, the specified value will replace the displaced value in the memory location. A carriage return terminates the command.

F Command (fill memory)

F'Low Address',High Address','Data'

Memory from 'Low Address' through 'High Address' is filled with 'data'. If 'High Address' is equal to or smaller than 'Low Address' on the 'Low Address' is changed.

M Command (move memory)

M'Low Address',High Address','Dest Address'

Data from 'Low Address' through 'High Address' are moved to memory beginning at 'Dest Address'. If 'High Address' is equal to or smaller than 'Low Address' only the byte at 'Low Address' is moved. If 'Dest Address' is between 'Low Address' and 'High Address' the data from 'Low Address' to 'Dest Address' is repeated to fill the destination field.

B Command (binary dump memory)

B'Low Address','High Address'

Data from 'Low Address' through 'High Address' are output to the logical punch device in a binary format compatible with mits paper tape format. 'High Address' must be equal to or greater than 'Low Address', with one exception: If 'High Address' is zero, an end-of-file-record is output specifying 'Low Address' as the entry point address

L Command (load memory, binary)

L'Bias Address'

Data in binary format are read from the logical reader device and stored in memory at the load address specified in the input file plus 'Bias Address'. When an end-of-file record is encountered control is transferred to the specified entry point, address of zero terminates loading and the monitor remains in control.

W Command (write memory, Hexadecimal)

W'Low Address','High Address'

Data from 'Low Address' through 'High Address' are output to the logical punch device in a hexadecimal format compatible with Intel paper tape format. 'High Address' must be equal to or greater than 'Low Address' with one exception: If 'High Address' is zero an end-of-file record is output specifying 'Low Address' as the entry point address.

R Command (read to memory, hexadecimal)

R'Bias Address'

Data in hexadecimal format are read from the logical reader device and stored in memory at the load address specified in the input file plus 'Bias Address'. When an end-of-file record is encountered control is transferred to the specified entry point address if it is nonzero. An entry point address of zero terminates loading and the monitor remains in control.

N Command (null output)

N

Sixty null bytes (OOH) are output to the logical punch device.

K Command (copy files)

K

Bytes are continously read from the logical reader device and output to the logical punch device. This process continues until manually interrupted, I.E., by resetting the system.

G Command (goto)

G'Address','Breakpoint 1','Breakpoint 2'

If 'Address' is specified, control is transferred to 'Address'. If 'Address' is not specified, control is transferred to the address of the last encountered breakpoint, after program status (CPU registers and flags) is restored.

If 'Breakpoint 1' or 'Breakpoint 2' is specified, breakpoints (RST 1) replace the bytes at corresponding addresses. These addresses must contain the first byte of an instruction. If breakpoints are specified, a jump instruction is stored at location 0008H to return control to the monitor when a breakpoint, or any RST 1 instruction is executed. At this point, the monitor will save the program status and restore the bytes replaced by any known breakpoints. The program counter in the saved program status is decremented, so that program execution may be resumed with the instruction formerly replaced by the breakpoint. Monitor commands may then be used to display/modify memory or CPU registers, etc.

When the monitor is entered normally, i.e. by other than breakpoint execution, recording of existing breakpoints is destroyed. Therefore, if breakpoints are set, but not executed before the monitor is re-entered, the contents of the bytes containing those breakpoints must be manually restored.

RST 1 instructions other than known breakpoints may be used as pseudo-breakpoints, subject to certain restrictions. The jump instruction must be stored at location 0008H by previously setting a normal breakpoint. RST 1 instructions other than known breakpoints may be executed through normal program execution (RST 1 stored as part executing program) or instruction jam (interrupt).

When such a RST 1 instruction is encountered, the monitor saves the program status and resets known breakpoints. However, the program counter in the saved program status is not decremented, so program execution may be resumed at the next instruction.

X Command (register display/modify)

X'Register'

Register contents as of the last encountered breakpoint are displayed. 'Register' may be specified as A,B,C,D,E,F (flags), H,L,M (H and L combined), P (program counter) or S (stack pointer). The registers are displayed, in the above order, beginning with specified 'Register'. After each register content is displayed, the operator may change it by supplying the new value followed by a space or comma. If no new value is entered the old value is retained and the next register is displayed. The command is terminated by a carriage return, or display/modification of register S.

If 'Register' is not specified, all registers are displayed without operator intervention.

C Command (hexadecimal arithmetic)

C'Operandl','Operand2'

The sum and difference of 'Operandl' and Operand2' are displayed in hexadecimal on the console device.

A Command (assign I/O devices)

A'Logical'='Physical'

Physical device 'Physical' is assigned to logical device 'Logical'. 'Logical' may be any of the four system logical devices, I.E., console, reader, punch, or list. Only the first character of the device name is required. 'Physical' may be 0,1,2, or 3. This option is not fully implemented due to the lack of I/O devices.

H Command (Help)

H

This program lists a summary of all of these commands.

T Command (Test Controller)

T

This command executes the test controller and test directory. The test directory printout can be suppessed by raising sense switch '0'. If the type-out is not suppressed the program will list the tests available and request the test to be run. If the type-out is suppressed, the test code can be input immediately following the 'T'.

A	Assigns I/O device (physical to logical)
B	Dump memory in binary on punch device
C	Hexadecimal arithmetic
D	Display a block of memory
F	Fill a block of memory with a constant
G	Go to address and execute, optional break-pt.
H	Help, this directory
K	Copy from reader to punch
L	Load a binary tape, optional bias
M	Move a block of memory to another location
N	Outputs 60 nulls to punch
R	Loads a hex tape from reader
S	Display and changes any memory location
T	Test list and execution program
W	Dumps memory in hex on punch
X	Cpu register display and change

Table 4.1 COMMAND SUMMARY

4.5 Externally Referenced Subroutines

Several externally reference subroutines are available for program usage. These routines, their starting address, and function are outlined below:

MONTRA 'F000'

Normal entry point to the monitor

CI 'F003'

Console input. One character is read from the logical console device and returned in register A. All registers other than A and F are preserved.

RI 'F006'

Reader input. One byte is read from the logical reader device and returned in reg A. All registers other than A and F are preserved. If no byte is available from the reader, the carry flag is set upon return.

CO 'F009'

Console output. The byte in register C is output to the logical console device. All registers other than A and F are preserved.

PO 'F00C'

Punch output. The byte in register C is output to the logical punch device. All registers other than A and F are preserved.

LO 'F00F'

List output. The byte in register C is output to the logical list device. All registers other than A and F are preserved.

CSTS 'F012'

Console status. The logical console input device is checked for availability. Register A is set to zero and the zero flag is set true if no input is available. Register A is set non-zero and the zero flag set false if a character is available. All registers other than A and F are preserved.

IOCHK 'F015'

The current setting of IOBYT (I/O byte) is returned in register A. IOBYT is the byte of ram used to record the current logical device to physical device assignments.

Bits 0,1 Record the physical device currently assigned to the logical console device.

Bits 2,3 Record the physical device currently assigned to the logical reader device.

Bits 4,5 Record the physical device currently assigned to the logical punch device.

Bits 6,7 Record the physical device currently assigned to the logical list device.

IOSET 'F018'

The contents of register C are stored in ICBYT, thus altering the logical to physical device assignments. All registers are preserved.

STRNG 'F01E'

The string of characters pointed to by registers H and L is output to the logical console device. The character string is terminated before a null character or after a character with bit 7 set. Registers B,D,E are preserved.

REENT 'F021'

Alternate entry point to the monitor. The current I/O configuration is not altered when the monitor is entered at this point.

Section 5

Mini-Memory Test

5.1 General Description

The Mini-Memory test is a small memory diagnostic test. The test is completely self-contained and requires no scratch pad memory or I/O devices. Since the test has a fixed test address range three different copies of the test are provided, each with a different address range.

5.2 Program Details

The Mini-Memory test is a modified implementation of the memory test supplied with the IMSAI 8080 computer system. To proide flexibility three different versions of the test are provided with a (0 to 256), (0 to 8K), (0 to24K) address ranges. All versions are stored in programmable memory on the diagnostic memory board.

The memory test consists of three phases. Phase one consists of an incremented bit pattern, where each address is tested with the 256 different patterns. In phase two and three the lower and upper bytes respectively of the address are stored in that location. Phase two and three are designed to help locate addressing problems.

5.3 Cperation

The 0 to 100 hex version of this test was designed to test the scratch pad area used by the dianostic operating system and the comprehensive memory test. This test should be run before these programs to verify this area of memory. Although this test can be run by the operating system test controller, it should normally be started directly from the IMSAI 8080 front panel at a starting address of 'C290' hex.

The Mini-Memory tests require no console device so all communication with the test is through the IMSAI 8080 sense lites (address lites 8-15) and the sense switches (address switches 8-15). Once the test is started, the status of the test as it proceeds through the various phases is displayed in the sense lites (see table 5.1). If an error is encounted, an error message is also read out in the sense lites. The following is the procedure used to locate the faulty memory location:

1. Change any sense switch
2. Sense lites will display 8 high-order address bits at at the failing location.
3. Change any sense switch.
4. Sense lites will display 8 low-order address bits at the failing location
5. Change any sense switch.
6. Sense lites will display data test pattern.

7. Change any sense switch.

8. Sense lites will display the actual data at the failing location.

9. Change any sense switch.

10. The test will start over at the beginning of phase one.

The 0 to 8K version of the Mini-Memory begins at 'D600' hex and the 0 to 24K version begins at 'D700' hex.

Sense Lite Display Hex	Meaning
01	Phase 1 Running-no errors yet
02	Phase 2 Running-no errors yet
03	Phase 3 Running-no errors yet
F1	Error Phase 1
F2	Error Phase 2
F3	Error Phase 3
FF	Test complete-no errors, move any sense switch to restart

Table 5.1

Phase messages

```
;  MINI-MEMORY TEST
;PROM VERSION FOR  0 TO 100H
;
;  BRIAN J. DONLAN
;
;
        ORG     0C290H
ENTER:  DI
        MVI     A,0FEH
        OUT     0FFH            ;OUTPUT PHASE I LITES
        LXI     H,000H          ;START ADDRESS
LP2:    XRA     A               ;ZERO ACC
LP1:    MOV     M,A             ;STORE TEST PATTERN IN MEM.
        MOV     B,M             ;READ BACK TO B
        CMP     B               ;COMPARE FOR OK
        JNZ     ERR1            ;JUMP IF ERROR
        INR     A               ;NEW TEST PATTERN
        JNZ     LP1
        INX     H
        LXI     D,0FF00H                ;STOP ADDRESS
        XCHG
        DAD     D               ;ADD TWO'S COMPLIMENT
        XCHG
        JNC     LP2
;
; PHASE II
        MVI     A,0FDH          ;PHASE II LITES
        OUT     0FFH
        LXI     H,000H
LP3:    MOV     M,H             ;LOW ADDRESS TO MEM
        INX     H
        LXI     D,0FF00H        ;STOP ADDRESS
        XCHG
        DAD     D
        XCHG
        JNC     LP3
;       READ MEMORY
        LXI     H,000H
LP4:    MOV     A,M             ;READ MEMORY
        SUB     H               ;COMPARE
        JNZ     ERR2            ;JUMP IF ERROR
        INX     H
        LXI     D,0FF00H
        XCHG
        DAD     D
        XCHG
        JNC     LP4
;
; PHASE III
;
        MVI     A,0FCH
        OUT     0FFH            ;PHASE THREE LITES
        LXI     H,000H
LP5:    MOV     M,L             ;STORE HIGH ADDRESS IN ALL  MEM
        INX     H
        LXI     D,0FF00H
        XCHG
        DAD     D
        XCHG
        JNC     LP5
;READ MEM
        LXI     H,000H
LP6:    MOV     A,M             ;READ MEMORY
        SUB     L               ;COMPARE
        JNZ     ERR3
        INX     H
        LXI     D,0FF00H
        XCHG
        DAD     D
        XCHG
        JNC     LP6
```

```
;
;
;    ALL PHASE COMPLETE
        MVI     A,0FFH
        LXI     H,ENTER
        JMP     LITES                   ;GO TO LITES PROG
;
;
;PHASE I ERROR
ERR1:   XCHG
        MOV     C,A                     ;SAVE BAD DATA
        LXI     H,COMERR                        ;RETURN
        MVI     A,0F1H                  ;PHASE I ERROR LITES
        JMP     LITES
;
;
;COMMON ERROR OUTPUT ROUTINE
COMERR: MOV     A,D                     ;HIGH ADDRESS
        LXI     H,LOADD                 ;RETURN
        JMP     LITES
LOADD:  MOV     A,E                     ;LOW ADDRES TO LITES
        LXI     H,TPAT                  ;RETURN
        JMP     LITES
TPAT:   MOV     A,C                     ;TEST PATTERN TO LITES
        LXI     H,ACTDAT                ;RETURN
        JMP     LITES
ACTDAT: MOV     A,B                     ;ACTUAL DATA TO LITES
        LXI     H,ENTER                         ;START OVER
        JMP     LITES
;;
;
; PHASE II ERROR
ERR2:   XCHG                            ;SAVE BAD ADDRESS
        ADD     D
        MOV     B,A
        MOV     C,D
        MVI     A,0F2H                  ;PHASE II ERROR TO LITES
        LXI     H,COMERR                        ;RETURN
        JMP     LITES
;
;
; PHASE III ERROR
ERR3:   XCHG                ;SAVE BAD ADDRESS
        ADD     E
        MOV     B,A
        MOV     C,E
        MVI     A,0F3H                  ;PHASE II ERRO TO LITES
        LXI     H,COMERR                ;RETURN
        JMP     LITES
;
;
;LITES ROUTINE     ENTER WITH RETURN IN REG H&L
;                             DATA FOR LITES IN A
LITES:  CMA
        OUT     0FFH                    ;OUTPUT LITES
        SPHL                            ;SAVE RETURN IN SP
        IN      0FFH                    ;READ SENSE SWITCHES
        MOV     H,A                     ;SAVE IN H
LP7:    IN      0FFH                    ;READ SWITCHES
        XRA     H                       ;SEE IF THEY CHANGED
        JZ      LP7
        LXI     H,0FC18H
LP8:    INX     H                               ;DELAY LOOP
        XRA     A
        ORA     H
        JNZ     LP8
        LXI     H,0                     ;ZERO H
        DAD     SP                      ;MOVE RETURN BACK TO H & L
        PCHL                            ;RETURN
```

```
;8K MINI MEMORY TEST
;
;
; BRIAN DONLAN
; PROM VERSION
        ORG     OD600H
ENTER:  DI
        MVI     A,OFEH
        OUT     OFFH                    ;OUTPUT PHASE I LITES
        LXI     H,000H                  ;START ADDRESS
LP2:    XRA     A                       ;ZERO ACC
LP1:    MOV     M,A                     ;STORE TEST PATTERN IN MEM.
        MOV     B,M                     ;READ BACK TO B
        CMP     B                       ;COMPARE FOR OK
        JNZ     ERR1                    ;JUMP IF ERROR
        INR     A                       ;NEW TEST PATTERN
        JNZ     LP1
        INX     H
        LXI     D,0E000H                        ;STOP ADDRESS
        XCHG
        DAD     D                       ;ADD TWO'S COMPLIMENT
        XCHG
        JNC     LP2
;
; PHASE II
        MVI     A,OFDH                  ;PHASE II LITES
        OUT     OFFH
        LXI     H,000H
LP3:    MOV     M,H                     ;LOW ADDRESS TO MEM
        INX     H
        LXI     D,0E000H                ;STOP ADDRESS
        XCHG
        DAD     D
        XCHG
        JNC     LP3
;       READ MEMORY
        LXI     H,000H
LP4:    MOV     A,M                     ;READ MEMORY
        SUB     H                       ;COMPARE
        JNZ     ERR2                    ;JUMP IF ERROR
        INX     H
        LXI     D,0E000H
        XCHG
        DAD     D
        XCHG
        JNC     LP4
;
; PHASE III
;
        MVI     A,OFCH
        OUT     OFFH                    ;PHASE THREE LITES
        LXI     H,000H
LP5:    MOV     M,L                     ;STORE HIGH ADDRESS IN ALL  MEM
        INX     H
        LXI     D,0E000H
        XCHG
        DAD     D
        XCHG
        JNC     LP5
;READ MEM
        LXI     H,000H
LP6:    MOV     A,M                     ;READ MEMORY
        SUB     L                       ;COMPARE
        JNZ     ERR3
        INX     H
        LXI     D,0E000H
        XCHG
        DAD     D
        XCHG
        JNC     LP6
```

```
;    ALL PHASE COMPLETE
        MVI     A,OFFH
        LXI     H,ENTER
        JMP     LITES                   ;GO TO LITES PROG
;
;
;PHASE I ERROR
ERR1:   XCHG
        MOV     C,A                     ;SAVE BAD DATA
        LXI     H,COMERR                        ;RETURN
        MVI     A,OF1H                  ;PHASE I ERROR LITES
        JMP     LITES
;
;
;COMMON ERROR OUTPUT ROUTINE
COMERR: MOV     A,D                     ;HIGH ADDRESS
        LXI     H,LOADD                 ;RETURN
        JMP     LITES
LOADD:  MOV     A,E                     ;LOW ADDRES TO LITES
        LXI     H,TPAT                  ;RETURN
        JMP     LITES
TPAT:   MOV     A,C                     ;TEST PATTERN TO LITES
        LXI     H,ACTDAT                ;RETURN
        JMP     LITES
ACTDAT: MOV     A,B                     ;ACTUAL DATA TO LITES
        LXI     H,ENTER                         ;START OVER
        JMP     LITES
;;
;
; PHASE II ERROR
ERR2:   XCHG                            ;SAVE BAD ADDRESS
        ADD     D
        MOV     B,A
        MOV     C,D
        MVI     A,OF2H                  ;PHASE II ERROR TO LITES
        LXI     H,COMERR                        ;RETURN
        JMP     LITES
;
;
; PHASE III ERROR
ERR3:   XCHG                    ;SAVE BAD ADDRESS
        ADD     E
        MOV     B,A
        MOV     C,E
        MVI     A,OF3H                  ;PHASE II ERRO TO LITES
        LXI     H,COMERR                ;RETURN
        JMP     LITES
;
;
;LITES ROUTINE      ENTER WITH RETURN IN REG H&L
;                              DATA FOR LITES IN A
LITES:  CMA
        OUT     OFFH                    ;OUTPUT LITES
        SPHL                            ;SAVE RETURN IN SP
        IN      OFFH                    ;READ SENSE SWITCHES
        MOV     H,A                     ;SAVE IN H
LP7:    IN      OFFH                    ;READ SWITCHES
        XRA     H                       ;SEE IF THEY CHANGED
        JZ      LP7
        LXI     H,OFC18H                        ;DELAY LOOP
LP8:    INX     H
        XRA     A
        ORA     H
        JNZ     LP8
        LXI     H,O                     ;ZERO H
        DAD     SP                      ;MOVE RETURN BACK TO H & L
        PCHL                            ;RETURN
```

```
 ; 24K MINI-MEMORY TEST
 ;  PROM VERSION
 ;
 ;
 ;
 ; BRIAN DONLAN
         ORG      0D700H
ENTER2:  DI
         MVI      A,0FEH
         OUT      0FFH               ;OUTPUT PHASE I LITES
         LXI      H,000H             ;START ADDRESS
LP22:    XRA      A                  ;ZERO ACC
LP12:    MOV      M,A                ;STORE TEST PATTERN IN MEM.
         MOV      B,M                ;READ BACK TO B
         CMP      B                  ;COMPARE FOR OK
         JNZ      ERR12              ;JUMP IF ERROR
         INR      A                  ;NEW TEST PATTERN
         JNZ      LP12
         INX      H
         LXI      D,0A000H
         XCHG                                ;STOP ADDRESS
         DAD      D                  ;ADD TWO'S COMPLIMENT
         XCHG
         JNC      LP22
 ;
 ; PHASE II
         MVI      A,0FDH             ;PHASE II LITES
         OUT      0FFH
         LXI      H,000H
LP32:    MOV      M,H                ;LOW ADDRESS TO MEM
         INX      H
         LXI      D,0A000H           ;STOP ADDRESS
         XCHG
         DAD      D
         XCHG
         JNC      LP32
 ;       READ MEMORY
         LXI      H,000H
LP42:    MOV      A,M                ;READ MEMORY
         SUB      H                  ;COMPARE
         JNZ      ERR22              ;JUMP IF ERROR
         INX      H
         LXI      D,0A000H
         XCHG
         DAD      D
         XCHG
         JNC      LP42
 ;
 ; PHASE III
 ;
         MVI      A,0FCH
         OUT      0FFH               ;PHASE THREE LITES
         LXI      H,000H
LP52:    MOV      M,L                ;STORE HIGH ADDRESS IN ALL  MEM
         INX      H
         LXI      D,0A000H
         XCHG
         DAD      D
         XCHG
         JNC      LP52
;READ MEM
         LXI      H,000H
LP62:    MOV      A,M                ;READ MEMORY
         SUB      L                  ;COMPARE
         JNZ      ERR32
         INX      H
         LXI      D,0A000H
         XCHG
         DAD      D
         XCHG
         JNC      LP62
```

```
;
;   ALL PHASE COMPLETE
        MVI     A,OFFH
        LXI     H,ENTER2
        JMP     LITES2              ;GO TO LITES PROG
;
;
;PHASE I ERROR
ERR12:  XCHG
        MOV     C,A                 ;SAVE BAD DATA
        LXI     H,COMER2                  ;RETURN
        MVI     A,OF1H              ;PHASE I ERROR LITES
        JMP     LITES2
;
;
;COMMON ERROR OUTPUT ROUTINE
COMER2: MOV     A,D                 ;HIGH ADDRESS
        LXI     H,LOADD2                  ;RETURN
        JMP     LITES2
LOADD2: MOV     A,E                 ;LOW ADDRES TO LITES
        LXI     H,TPAT2             ;RETURN
        JMP     LITES2
TPAT2:  MOV     A,C                 ;TEST PATTERN TO LITES
        LXI     H,ACTDA2
        JMP     LITES2
ACTDA2: MOV     A,B                 ;ACTUAL DATA TO LITES
        LXI     H,ENTER2                          ;START OVER
        JMP     LITES2
;;
;
; PHASE II ERROR
ERR22:  XCHG                        ;SAVE BAD ADDRESS
        ADD     D
        MOV     B,A
        MOV     C,D
        MVI     A,OF2H              ;PHASE II ERROR TO LITES
        LXI     H,COMER2                  ;RETURN
        JMP     LITES2
;
;
; PHASE III ERROR
ERR32:  XCHG                ;SAVE BAD ADDRESS
        ADD     E
        MOV     B,A
        MOV     C,E
        MVI     A,OF3H              ;PHASE II ERRO TO LITES
        LXI     H,COMER2            ;RETURN
        JMP     LITES2
;
;
;LITES ROUTINE     ENTER WITH RETURN IN REG H&L
;                             DATA FOR LITES IN A
LITES2: CMA
        OUT     OFFH                ;OUTPUT LITES
        SPHL                        ;SAVE RETURN IN SP
        IN      OFFH                ;READ SENSE SWITCHES
        MOV     H,A                 ;SAVE IN H
LP72:   IN      OFFH                ;READ SWITCHES
        XRA     H                   ;SEE IF THEY CHANGED
        JZ      LP72
        LXI     H,OFC18H
LP82:   INX     H                          ;DELAY LOOP
        XRA     A
        ORA     H
        JNZ     LP82
        LXI     H,O                 ;ZERO H
        DAD     SP                  ;MOVE RETURN BACK TO H & L
        PCHL                        ;RETURN
```

Section 6

Comprehensive Memory Test

6.1 General Description

This memory test is a comprehensive memory diagnostic. The program will test the read, write, data hold and addressing capabilities of a block of memory between any two given locations.

6.2 Program Details

The comprehensive memory test is based on an implementation by Bock W. Lee [1] of the K and H memory test algorithm. The K and H algorithm, named after its creators, J. Knaizuk and C. Hartmann [2] uses modulo three addressing to help addressing problems which might normally be hidden.

The program uses two test patterns which are compliments of each other, the major and minor pattern. First one and then the other pattern are written in every third location and then read back as the program cycles through all memory locations. After all memory locations are tested using the major and minor patterns, a pass is complete. The patterns are then rotated and another pass through memory begins with the new test patterns.

6.3 Operation

The user is required to input the starting and stopping addresses. The test will continue to cycle through the test using the cyclic patterns until it is interrupted by typing any console character.

The test has error messages which indicate the failing location and the test pattern which fails. End of pass messages are also given after each complete pass with one pair of test patterns.

The memory test is stored in programmable read-only memory on the diagnostic memory board. The program can be run under the diagnostic operating system test controller, or it can be started from the IMSAI 8080 front panel at a starting address of 'C000'.

The memory test is also stored on floppy disk and can be invoked by the CPM operating system under file name 'MEMTST.COM'

Note: The memory test requires the first 100 hex address for scratch pad so these addresses should not be tested using this test. The first 100 addresses should be tested using the mini-memory test.

```
;         MEMORY TEST
;  DISC VERSION  24 MAY 79  B. DONLAN
;
          ORG       100H
;
;
WO        EQU       00        ;TEST BYTE
;
ENTRY1:   LXI       H,OFOH
          SPHL
ENTRY:    LXI       H,ENTRY
          PUSH      H
          MVI       A,00                ;ZERO ACC
          STA       CODE
          CALL      CRLF
          LXI       H,MSG1
          CALL      PMSG
          LXI       H,MSG2
          CALL      PMSG
          CALL      BBIN
          XCHG
          SHLD      START
          LXI       H,MSG3
          CALL      PMSG
          CALL      BBIN
          XCHG
          SHLD      ENADR
          LXI       H,MSG8
          CALL      PMSG
          IN        CDATA               ;RESET IO FLAG
;
;
BEGIN:    MVI       C,WO      ;LOAD TEST BYTE
MTEST:    MVI       A,02      ;LOAD TEST BYTE
          STA        PART
MTLOP:    CALL      STUFF     ;STUFF MAJOR ALL OVER
          MVI       A,02      ;SET TWO AS MINOR
          CALL      STUFM     ;STUFF MINOR
          MVI       A,02      ; SET 2 AGAIN
          CALL      CHECK     ;NOW CHECK ALL LOC
PTCHK:    LDA       PART
          DCR       A
          STA       PART      ;STORE NEW PART
          CPI       00        ;FINISH THIS PASS ?
          JZ        RECYCLE   ;YES
CONT:     MVI       A,01      ;NO CONTINUE
          CALL      STUFM     ;STUFF MINOR SERT
          MOV       A,C       ;LOAD MAJOR BYTE
          CMA                 ;COMPLIMENT MAJOR BYTE
          MOV       C,A       ;SAVE NEW BYTE
          XRA       A         ;ZERO OTHER TEST BYTE
          CALL      CHECK
          JMP       MTLOP
;
;
RECYCLE:
          MOV       A,C
          STA       PART      ;SAVE INVERT TB TEMP
          LXI       H,MSG4    ;END OF PAS MESSAGE
          CALL      PMSG
          LDA       CODE      ;CHAR CODE
          ORA       A         ;SET FLAGS
          JNZ       ENTRY     ;START OVER
          ANA       A         ;CLEAR CARRY
          LDA       PART      ; RECOVER TEST BYTE
          ORA       A
          JZ        BEGIN
          RAL
          CMA
TB:       MOV       C,A       ;NEW TEST BYTE
          JMP       MTEST     ; ANOTHER PAS
```

```
,
START:   DS      2                       ;LOC FOR START ADDR
ENADR:   DS      2                       ;LOC FOR END ADDRESS
PART:    DS      1
CODE:    DS      1
;
;
STUFF:   CALL    STASTO  ;LOAD START AND END ADDR
DOIT:    MOV     M,C     ;STUFF MAJOR ALL OVER
         CALL    HILOX   ;SEE IF ALL MEM DONE
         JMP     DOIT    ;NO KEPP ON STUFFING
;
;
;
STUFM:   CALL    STASTO  ;LOAD ADDR AGAIN
         MOV     B,A     ;MINOR COUNTER
         CPI     00      ;MINOR WORD STUFF
         JNZ     HIL     ;NO
MINOR:   MOV     A,C     ;MAJOR TEST BYTE
         CMA             ;MINOR IS COMPLIMENT OF MAJOR
         MOV     M,A     ;STUFF MINOR BYTE IN MEM
         MVI     B,03    ;START MINOR COUNT AT 3
HIL:     CALL    HILOX   ;INC & CHK IF DONE
         DCR     B       ;DEC MINOR COUNTER
         JNZ     HIL     ;OK TO STUFF NO
         JMP     MINOR   ;YES
;
;
CHECK:   CALL    STASTO          ;LOAD START AND END
         MOV     B,A     ;LOAD MINOR COUNT
         CPI     00      ;COUNT ZERO
         JNZ     MAJR    ;NO GO TO MAJOR
MINR:    MOV     A,C     ;LOAD TEST BYTE MAJOR
         CMA             ; MINOR IS COMPLIMENYT
         CMP     M       ;READ AND COMPARE MEM LOC
         MVI     B,03    ;MINOR COUNT AT 3
         JMP     CKEND   ;CHECK FOR ERROR OR ABORT
MAJR:    MOV     A,C     ;LOAD MAJOR TEST BYTE
         CMP     M       ;READ AND COMPARE MEM WITH MAJOR
CKEND:   PUSH    B       ;SAVE COUNT AND MAJOR
         CNZ     ERR     ;GO TO ERR TO PRNT IF ERROR
         POP     B       ;RESTORE REGS
         IN      CSTAT   ;CHECK KEYBOARD
         ANI     02H
         JZ      FIN
         IN      CDATA   ;READ KEYS
         STA     CODE
FIN:     CALL    HILOX
         DCR     B       ;DEC MINOR COUNT
         JNZ     MAJR
         JMP     MINR    ;COUNT ZERO DO MINOR
;
;
;
;
STASTO:  LHLD    ENADR   ;LOAD END ADDR
         XCHG            ;MOVE END TO C&D
         LHLD    START   ;LOAD START
         RET
;
;
ERR:     PUSH    D       ;SAVE END ADDR
         PUSH    PSW
         CALL    CRLF
         MOV     D,H
         MOV     E,L
         CALL    BINB    ;OUTPUT BAD ADDR
         MVI     D,08    ;SPACE COUNT
         CALL    BLNK    ;SPACE OVER 8
         POP     PSW
         MOV     B,A
         CALL    BITS    ;PRINT TEST BYTE
         MVI     D,OAH
         CALL    BLNK
         MOV     A,M
         CALL    BITS    ;PRINT BAD BYTE
         MOV     A,B     ;MOVE TEST BYTE BACK
         POP     D       ;RESTORE END ADDR
         RET
```

```
HILOX:  PUSH    A       ;SAVE ACC
        INX     H       ;INC CURRENT ADDR
        MOV     A,H     ;LOAD HIGH ODER ADDR
        CMP     D       ;COMPARE WITH END
        JNZ     DIFF    ;NO MATCH
        MOV     A,L     ;LOAD LOW ORDER
        CMP     E       ;COMPARE LOW ORDERS
        JNZ     DIFF    ;NO MATCH
        POP     A       ;MATCH END
        INX     SP      ;FAKE RETURN ONE LEVEL OUT
        INX     SP
        RET
DIFF:   POP     A       ;CONTINUE STUFFING
        RET
PROB:   MVI     C,3FH   ; ?
        CALL    CONOT   ;PRINT ?
        JMP     ENTRY
;
;
MSG1    DB      ODH,OAH,'MEMORY TEST',0
MSG2    DB      ODH,OAH,'ENTER START ADDRESS     ',0
MSG3    DB      ODH,OAH,'ENTER STOP ADDRESS     ',0
MSG4    DB      ODH,OAH,'END OF PASS      ',OAH,0
; diagnostic input output routines
; for brian donlan  26 feb 79

CSTAT   EQU  3          ;CONSOLE STATUS PORT.
CCOM    EQU  3          ;CONSOLE COMMAND PORT.
CDATA   EQU  2          ;CONSOLE DATA PORT.
CKBR    EQU  00000010B  ;KEYBOARD READY BIT.
CPTR    EQU  00000001B  ;PRINT READY BIT.
CNULL   EQU  1          ;CONSOLE NULL COUNT.


; CHECK CONSOLE INPUT STATUS.
;

CONST:  IN   CSTAT      ;READ CONSOLE STATUS.
        ANI  CKBR       ;LOOK AT KB READY BIT.
        MVI  A,0        ;SET A=0 FOR RETURN.
        RZ              ;NOT READY WHEN ZERO.
        CMA             ;IF READY A=FF.
        RET             ;RETURN FROM CONST.

;
; READ A CHARACTER FROM CONSOLE.
;
CONIN:   IN   CSTAT             ;READ CONSOLE STATUS.
         ANI  CKBR      ;IF NOT READY,

        JZ   CONIN      ;READY WHEN HIGH.
         IN   CDATA     ;READ A CHARACTER.
        OUT     CDATA
         ANI  7FH       ;MAKE MOST SIG. BIT = 0.
         RET
;
; WRITE A CHARACTER TO THE CONSOLE DEVICE.
;
CONOT:  MVI  A,ODH      ;IF IT'S A CR,
        CMP  C          ;THEN HOP OUT
        JZ   CONUL      ;TO NULL ROUTINE.
CONOT1: IN   CSTAT      ;READ CONSOLE STATUS.
        ANI  CPTR       ;IF NOT READY,
        JZ   CONOT1     ;READY WHEN HIGH.
        MOV  A,C        ;GET CHARACTER.
        OUT  CDATA      ;PRINT IT.
        RET             ;RETURN.
CONUL:  PUSH B          ;SAVE B&C.
        MVI  B,CNULL    ;GET NULL COUNT.
CONUL1: CALL CONOT1     ;PRINT CR.
        MVI  C,0        ;GET NULL CHAR.
        DCR  B          ;DECREMENT COUNTER.
        JNZ  CONUL1     ;DO NEXT NULL.
        POP  B          ;RESTORE B&C.
        MOV  A,C        ;RESTORE A.
        RET             ;RETURN.
```

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;       PRINT MESSAGE UNTIL ZERO
;       MESSAGE ADDRESS REG H & L
;
;;;;;;;;;;;;;;;;;;;;;
PMSG:   MOV     A,M     ;GET CHAR
        ORA     A       ;IS IT A  ZERO
        RZ
        MOV     C,A     ;OTHERWISE PRINT
        CALL    CONOT
        INX     H       ;INC ADDRESSS
        JMP     PMSG
;
;
;
MSG8    DB      ODH,OAH,OAH,'LOC.        TEST BYTE        MEMORY BYTE',0
;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;    PRINT 8 BIT WORD IN BINARY FORMAT
;       INPUT:  DATA IN REG A
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;
BITS:   MOV     B,A     ; DATA
        MVI     A,80H   ; MASK
OVER:   MVI     C,30H
        MOV     E,A     ; STORE MASK
        ANA     B       ; AND WITH MASK
        JZ      PRNT    ; JUMP IF ZERO
        MVI     C,31H
PRNT:   CALL    CONOT
        ANA     B       ; ZERO CARRY
        MOV     A,E     ; LOAD MASK
        RAR
        JNC     OVER
        RET
;;
;
BLNK:   MVI     C,20H           ;PRINT BLANKS, # IN REG. D
LP1:    CALL    CONOT1
        DCR     D
        JNZ     LP1
        RET
;
BINHA:  MOV     A,D
        RAR
        RAR
        RAR
        RAR
        CALL    BIN1
        MOV     C,A
        CALL    CONOT
        MOV     A,D
        CALL    BIN1
        MOV     C,A
        CALL    CONOT
        RET
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;    OUTPUTS FOUR HEX DIGITS IN ASCII
;    ENTER WITH DATA IN REG PAIR E AND D
;;;;;;;;;;;;;;;;;;
;;
;
BINB:   CALL    BINHA
        MOV     D,E
        CALL    BINHA
        RET
```

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;    CONVERTS HEX TO ASCII
;    INPUT: 4 BITS HEX  REG A
;    OUTPUT:  8 BIT ASSCII REG A
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;
;
BIN1:   ANI     0FH
        ADI     30H
        CPI     3AH
        RC
        ADI     07H
        RET
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;    INPUTS 4 DIGITS FROM CONSOLE
;    RETURN;  4 HEX DIGITS IN REG E-D
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;
;
BBIN:   CALL    CONIN
        CALL    AHS1
        RAL
        RAL
        RAL
        RAL
        ANI     0F0H
        MOV     D,A
        CALL    CONIN
        CALL    AHS1
        ANI     0FH
        ORA     D
        MOV     D,A
        CALL    CONIN
        CALL    AHS1
        RAL
        RAL
        RAL
        RAL
        ANI     0F0H
        MOV     E,A
        CALL    CONIN
        CALL    AHS1
        ANI     0FH
        ORA     E
        MOV     E,A
        RET
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;    CONVERT ASCII TO HEX
;    INPUT;  8 BIT ASCII  REG A
;    OUTPUT:  4 BIT HEX REG A
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;
;
AHS1:   NOP
        SUI     30H
        CPI     0AH
        RC
        SUI     07H
        RET
```

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;     INITIATE SIO PORTS
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;
;
INITA:  MVI     A,0AAH          ;GET DUMMY MODE WORD
         OUT    CSTAT           ;OUTPUT IT
        MVI     A,40H           ;GET RESET BIT
        OUT     CSTAT           ;RESET SIO BOARD
        MVI     A,0CEH          ;GET REAL MODE WORD
        OUT     CSTAT           ;SET THE MODE FOR REAL
        MVI     A,37H           ;GET THE COMMAND
        OUT     CSTAT           ;OUTPUT IT
        RET
;
;
;
CRLF:   MVI     C,13            ;CR
        CALL    CONOT
LF:     MVI     C,10            ;LF
        CALL    CONOT1
        MVI     C,7FH
        CALL    CONOT1
        CALL    CONOT
        RET
```

Section 7

Formatted Disk Test

7.1 General Description

The formatted disk test is designed to test the operation of the Pertec floppy disk drive and the Tarbell Disk controller. The ability of the disk system to read, write and seek tracks is tested in the normal formatted mode.

7.2 Program Details

The formatted disk test is the largest and most complex of all the diagnostics in this package. The test is completely self-contained and requires no external I/O subroutines.

The ability of the disk system to read, write and seek tracks is tested by writing a known test pattern and then repositioning the read/write head before performing a verification read. In order to test the disk drive for track positioning and skew error the head is moved between each read and write. The test sequence is as follows:

1. write inner track
2. seek outer track
3. write outer track
4. seek inner track
5. read and verify inner track
6. seek outer track
7. read and verify outer track
8. increment inner and outer track counters

The inner track starts at one and the outer track starts at 38. This continues until all 26 sectors on 77 tracks are tested. Extensive error checking is performed on both the read/write data and the disk status. A large number number of error messages are provided to aid in error analysis. All seek error and read/write data error messages include the sector and track number in question.

7.3 Operation

This test will request that a formatted scratch disk for reading and writing be mounted. The mounting of the disk must then be confirmed by the operator typing a 'Y' on the console device.

The test requires no further interaction unless an error is encountered. After an error is reported, the operator must instruct the program whether to repeat(R) the last sector test or to continue(C) on to the next sector. Raising sense switch '0' will direct the test to automatically continue after an error.

The test can be stopped at any time by typing a 'control B' on the console device.

Each sector contains 128 data bytes. When a read data verification error is encountered, the faulty track and sector are reported and the number of incorrect bytes in the sector is counted. Only the last errant data byte is listed.

The disk test is stored in programmable memory on the diagnostic memory board. The disk test can be run under the diagnostic operating system test controller, or it can be started from the IMSAI 8080 front panel at a starting address of 'C800'.

The disk test is also stored on floppy disk and it can be invoked by the CPM operating system under file name 'DSKTST.COM'.

```
;       DISK TEST FOR TARBELL DISK CONTROLLER
;       BRIAN J. DONLAN
;       18  MAR  79
;   DISC VERSION
;
;
        ORG     0100H
ENTRY:  LXI     H,MSG1          ;OPENING MESSAGE
        CALL    PMSG
        LXI     H,MSG1A
        CALL    PMSG
        CALL    CONIN           ;CHECK KEYBOARD
        CPI     'Y'             ;CHECK IF Y
        JNZ     ENTRY           ;  ?? START OVER
        CALL    CRLF
LOOP6:  CALL    CRLF
        XRA     A               ;ZERO ACC
        STA     ERRFLG          ;ZERO EERROR FLAG
        STA     LPCNT           ;ZERO LOOP COUNT
        CALL    HOME            ;HOME DRIVE TO TRK 0
LOOP4:  XRA     A
        STA     INNER           ;ZERO INNER TRK
        MVI     A,38            ;OUTER TRK
        STA     OUTER
        CALL    PAT             ;GET PATTERN
        CALL    INWRT
        MVI     A,34
        CALL    SEEK            ;
        CALL    INRD            ;MOVE BACK AND CHECK TRK 00
        MVI     A,01            ;SET UP TO DO PAIRS
        STA     INNER           ;START PAIRS WITH TRKO1
;
; TEST FOR CONSOLE INTERRUPT
LOOP8:  IN      CSTAT
        ANI     02H             ;KEYBAORD READY
        JZ      LOOP3           ;NO
        IN      CDATA           ;READ KEYS
        CPI     03H             ;CONTROL C
        CPI     02H             ;CONTROLB
        JZ      ENTRY           ;START OVER AGAIN
        ;
LOOP3:  CALL    INWRT           ;WRITE INNER TRK
        CALL    OUTWRT
        CALL    INRD            ;READ INNER TRK
        CALL    OUTRD
        LDA     INNER
        INR     A
        STA     INNER
        ADI     38              ;FIND NEXT OUTER T4K
        STA     OUTER           ;STORE OUTER TRK
        CPI     77              ;TRK    77 YET ?
        JNZ     LOOP8           ;NOT DONE YET
        LDA     LPCNT           ;LOOP COUNTER
        INR     A
        STA     LPCNT
        JMP     LOOP4
;
;
;       PATTERN ROUNTINE EXPANDABLE
PAT:    LDA     LPCNT           ;LOAD LOOP COUNTER
        JZ      IST
        CPI     01              ;SECOND PASS
        JZ      SECD
        CPI     02
        JZ      THIRD
        LXI     H,MSG2          ;END OF PASS
        CALL    PMSG
        IN      CSTAT           ;CHECK KEYBOARD
        ANI     02H
        JZ      LOOP6
        HLT                     ;CONTINUE TEST UNTIL INTERUPTED
        JMP     ENTRY
IST:    MVI     A,OFFH          ;ALL ONES PATERN
        STA     PATEN           ;STORE PATTERN
        RET
SECD:   MVI     A,00H           ;ALL ZERO PATTERN
        STA     PATEN
        RET
THIRD   MVI     A,55H           ;ALTER PATTERN
        STA     PATEN
        RET
```

```
;
;        WRITE INNER TRK
INWRT:   LDA     INNER
         STA     TRK
BOTH:    CALL    SEEK             ;MOVE HEAD TO TRK
         MVI     A,01             ;FIRST SECTOR
         STA     SECT
;
LOOP1:   XRA     A                ;ZERO ACC
         STA     REPETE           ;ZERO REPEAT FLAG
         CALL    WRITE            ;WRITE ONE SECTOR
         LDA     REPETE           ;LOAD REPEAT FLAG
         ORA     A                ;SET FLAGS
         JNZ     LOOP1            ;REPEAT SECTOR
         LDA     SECT
         INR     A                ;INC SECTOR
         STA     SECT
         CPI     27               ;ALL SECTOR DONE ?
         JNZ     LOOP1            ;NO
         RET
;
;
;        WRITE OUTER TRK
OUTWRT:  LDA     OUTER            ;LOAD OUTER TRK
         STA     TRK
         JMP     BOTH             ;COMMON WRITE ROUNTINE
;
;
;        READ INNER TRK
INRD:    LDA     INNER
         STA     TRK
BOTH2:   CALL    SEEK             ;MOVE HEAD TO TRK
         MVI     A,01             ;FIRST SECTOR
         STA     SECT             ;ZERO SECTOR
;
LOOP5:   XRA     A
         STA     ERRFLG           ;ZREO ERROR COUNT
         STA     REPETE
         CALL    READ             ;READ ONE SECTOR
         XRA     A                ;ZERO ACC
         STA     ERRFLG
         LDA     REPETE           ;REPEAT FLAG
         ORA     A                ;SET FLAGS
         JNZ     LOOP5
         LDA     SECT
         INR     A
         STA     SECT             ;NEXT SECTOR
         CPI     27               ;ALL SECTORS DONE ?
         JNZ     LOOP5            ;NO
         RET
;
;
;        READ OUTER TRK
OUTRD:   LDA     OUTER            ;OUTER TRK NO.
         STA     TRK
         JMP     BOTH2
;
ERRPNT:  LXI     H,MSG3           ;ERROR MESSAGE
         CALL    PMSG
         LDA     ERRFLG           ;ERROR COUNT
         MOV     D,A
         CALL    BINHA            ;PRINT  ERROR COUNT
         LXI     H,MSG4           ;HEADINGS
         CALL    PMSG
         MVI     D,03             ;SPACE OVER
         CALL    BLNK
         LDA     TRK              ;TRACK NO.
         MOV     D,A
         CALL    BINHA            ;PRINT TRACK NO.
         MVI     D,16             ;SPACE OVER
         CALL    BLNK
         LDA     SECT             ;SECTOR NO.
         MOV     D,A
```

```
        CALL    BINHA           ;PRINT SECTOR NO.
        MVI     D,13            ;SPACE OVER
        CALL    BLNK
        LDA     PATEN
        CALL    BITS            ;PRINT TEST PATTERN
        MVI     D,12            ;SPACE OVE
        CALL    BLNK
        LDA     BADBT           ;LAST BAD BYTE
        CALL    BITS            ;PRINT LAST BAD BYTE
        RET
;
LPCNT:  DS      1               ;SPACE FOR LOOP COUNTER
INNER:  DS      1               ;SPACE FOR INNER TRK NO.
OUTER:  DS      1               ;SPACE FOR OUTER TRK NO.
PATEN   DS      1               ;SPACE FOR TEST PATTERN
ERRFLG: DS      1               ;SPACE FOR ERROR COUNT
BADBT:  DS      1               ;SPACE FOR BAD BYTE
BDTRK:  DS      1               ;SPACE FOR DISK READ TRK WHEN ERR
REPETE: DS      1               ;REPETE FLAG
MSG1:   DB      ODH,OAH,'DISK TEST NO. 1 FORMATTED TEST  ',0
MSG1A:  DB      ODH,OAH,'LOAD SCRATCH DISK  TYPE Y WHEN READY',0
MSG2    DB      ODH,OAH,' END OF PASS ',0
MSG3:   DB      ODH,OAH,'DATA ERROR ON DISK CHECK       ERROR COUNT IN HEX   ',0
MSG4:   DB      ODH,OAH,' TRACK NO.         SECTOR NO.       TEST BYTE         LAST ERROR'
MSG5:   DB      ODH,OAH,'HEAD POSITION ',0
MSG6:   DB      ODH,OAH,'DISK TRACK            CONTROLLER TRACK          SECTOR ',ODH,OAH,0
MSG7:   DB      ODH,OAH,ODH,OAH,'  !! EXECUTION STOPPED !! ',0
MSG8:   DB      ODH,OAH,'TYPE R TO RETRY,  C TO CONTINUE,  ANYTHING ELSE STOP ',0
;
CSTAT   EQU  3          ;CONSOLE STATUS PORT.
CCOM    EQU  3          ;CONSOLE COMMAND PORT.
CDATA   EQU  2          ;CONSOLE DATA PORT.
CKBR    EQU  00000010B  ;KEYBOARD READY BIT.
CPTR    EQU  00000001B  ;PRINT READY BIT.
CNULL   EQU  1          ;CONSOLE NULL COUNT.
DISK    EQU  OF8H       ;DISK BASE ADDRESS.
DCOM    EQU  DISK       ;DISK COMMAND PORT.
DSTAT   EQU  DISK       ;DISK STATUS PORT.
TRACK   EQU  DISK+1     ;DISK TRACK PORT.
SECTP   EQU  DISK+2     ;DISK SECTOR PORT.
DDATA   EQU  DISK+3     ;DISK DATA PORT.
WAIT    EQU  DISK+4     ;DISK WAIT PORT.
DCONT   EQU  DISK+4     ;DISK CONTROL PORT.

TRK:    DS      1               ;ADDRESS FOR TRACK
SECT:   DS      1               ;ADDRESS FOR SECTOR

;
; READ A CHARACTER FROM CONSOLE.
;
CONIN:   IN   CSTAT             ;READ CONSOLE STATUS.
         ANI  CKBR      ;IF NOT READY,

        JZ   CONIN      ;READY WHEN HIGH.
         IN   CDATA     ;READ A CHARACTER.
        OUT     CDATA
         ANI  7FH       ;MAKE MOST SIG. BIT = 0.
         RET

;
; WRITE A CHARACTER TO THE CONSOLE DEVICE.
;
CONOT:  MVI  A,ODH      ;IF IT'S A CR,
        CMP  C          ;THEN HOP OUT
        JZ   CONUL      ;TO NULL ROUTINE.
CONOT1: IN   CSTAT      ;READ CONSOLE STATUS.
        ANI  CPTR       ;IF NOT READY,
        JZ   CONOT1     ;READY WHEN HIGH.
        MOV  A,C        ;GET CHARACTER.
        OUT  CDATA      ;PRINT IT.
        RET             ;RETURN.
CONUL:  PUSH B          ;SAVE B&C.
        MVI  B,CNULL    ;GET NULL COUNT.
CONUL1: CALL CONOT1     ;PRINT CR.
        MVI  C,0        ;GET NULL CHAR.
        DCR  B          ;DECREMENT COUNTER.
        JNZ  CONUL1     ;DO NEXT NULL.
        POP  B          ;RESTORE B&C.
        MOV  A,C        ;RESTORE A.
        RET             ;RETURN.
```

```
; MOVE DISK TO TRACK ZERO.
;
HOME:   MVI  A,0D0H     ;CLEAR ANY PENDING COMMAND.
        OUT  DCOM
        XRA     A       ;ZERO ACC
        STA     TRK     ;STORE TRACK
HOME1:  IN   DSTAT      ;READ DISK STATUS.
        RRC             ;LOOK AT LSB.
        JC   HOME1      ;WAIT FOR NOT BUSY.
        MVI  A,3        ;20 MS STEP RATE.
        OUT  DCOM       ;ISSUE HOME COMMAND.
        IN   WAIT       ;WAIT FOR INTRQ.
        ORA  A          ;SET FLAGS.
        JM   HERR       ;ERROR IF DRQ.
        IN   DSTAT      ;READ DISK STATUS.
        MOV  D,A        ;SAVE IN REGISTER D.
        ANI  4          ;LOOK AT BIT 2.
        JZ   HERR       ;ERROR IF NOT TRK 0.
        MOV  A,D        ;GET STATUS BACK.
        ANI  91H        ;MASK NON-ERROR BITS.
        RZ              ;RETURN IF NO ERROR.
HERR:   LXI  H,HEMSG    ;PRINT "HOME ".
        MOV  A,D        ;MASK NON-ERROR BITS.
        ANI  91H
        MOV  D,A
        JMP  ERMSG      ;DO COMMON ERROR MSGS.
;
; SELECT DISK NUMBER.
;
INTDSK: MVI     A,02            ;DRIVE NO. 1
DSK1:   OUT  DCONT      ;SET THE LATCH WITH CODE.
         RET            ;RETURN FROM SELDSK.
;
; READ THE SECTOR AT SECT, FROM THE PRESENT TRACK.
;  SECTOR IN  SECT
;   HEAD LOAD FIRST
;
READ:   LXI     H,080H          ;READ BUFFER
        LDA     SECT
READ1:  OUT  SECTP      ;SET SECTOR INTO 1771.
        MVI  A,8CH      ;CODE FOR READ W/O HD LD.
READE:  OUT  DCOM       ;SEND COMMAND TO 1771.
RLOOP:  IN   WAIT       ;WAIT FOR DRQ OR INTRQ.
        ORA  A          ;SET FLAGS.
        JP   RDDONE     ;DONE IF INTRQ.
        IN   DDATA      ;READ A DATA BYTE FROM DISK.
;
        MOV     M,A             ;STORE IN BUFFER
        INX     H               ;INC BUFF POINTER
;
        JMP     RLOOP
;
; COMPARE DATA WITH TEST BYTE;
RDDONE: LXI     H,080H          ;HEAD OF BUFFER
        LDA     PATEN           ;TEST PATTERN
        MOV     B,A             ;PATTERN TO B
        MVI     D,080H          ;COUNTER FOR BYTES
COMPLP: MOV     A,M             ;GET DATA
        CMP     B               ;COMPARE WITH TB
        JNZ     DATERR          ;ERROR
ERRET:  INX     H
        DCR     D               ;DEC BYTE COUNT
        JNZ     COMPLP          ;DO 128 TIMES
        IN   DSTAT      ;READ DISK STATUS.
        ANI  9DH        ;LOOK AT ERROR BITS.
        MOV     D,A     ;SAVE ERROR BITS
        LDA     ERRFLG          ;READ ERROR FLAG
        ORA     D               ;SET FLAGS ON COMBO
        RZ              ;RETURN IF NONE.
        LXI  H,RDMSG    ;PRINT "READ ".
ERMSG:  CALL PMSG       ;PRINT ORIGIN MESSAGE.
```

```
;       COMMON ERROR PRINT OUT
;

ERMSG1: MOV  A,D          ;GET ERROR BITS.
        ANI  80H          ;IF BIT 7 HIGH,
        LXI  H,NRMSG      ;"NOT READY".
        CNZ PMSG
        MOV  A,D          ;GET ERROR BITS.
        ANI  10H          ;IF BIT 4 IS HIGH,
        LXI  H,RNMSG      ;PRINT "RECORD NOT FOUND"
        CNZ PMSG
        MOV  A,D          ;GET ERROR BITS.
        ANI  8H           ;IF BIT 3 IS HIGH,
        LXI  H,CRCMSG     ;PRINT "CRC ERROR".
        CNZ PMSG
        MOV  A,D          ;GET ERROR BITS.
        ANI  4H           ;IF BIT 2 IS HIGH,
        LXI  H,LDMSG      ;PRINT "LOST DATA".
        CNZ PMSG
        MOV  A,D          ;GET ERROR BITS.
        ANI  1            ;IF BIT 1 IS HIGH,
        LXI  H,BSYMSG     ;PRINT "BUSY".
        CNZ  PMSG
PERMSG: LXI  H,ERRMSG     ;PRINT "ERROR."
        CALL PMSG
        MOV      A,D              ;MOVE FLAGS TO ACC
        ANI      18H              ;CRC OR RECORD NOT FOUND
        JZ       RETRY
TRKCHK: MVI      A,0C4H
        OUT      DCOM             ;READ ADDRESS
        IN       WAIT
        IN       DDATA            ;TRACK ADDRESS
        STA      BDTRK
CHKS2   IN       WAIT             ;DUMP REST OF DATA
        JM       CHKS2
        LXI      H,MSG5           ;HEAD ERROR MESSAGE
        CALL     PMSG
        LXI      H,MSG6           ;HEADINGS
        CALL     PMSG
        MVI      D,05H
        CALL     BLNK             ;SPACE OVER
        LDA      BDTRK            ;DISK TRK
        MOV      D,A
        CALL     BINHA            ;PRINT TRK
        MVI      D,15H
        CALL     BLNK             ;SPACE OVER
        IN       TRACK
        MOV      D,A
        CALL     BINHA            ;PRINT TRK
        MVI      D,13H
        CALL     BLNK
        LDA      SECT             ;SECTOR
        MOV      D,A
        CALL     BINHA            ;PRINT SECTO NO.
RETRY:  LDA      ERRFLG
        ORA      A                ;SET FLAGS
        CNZ      ERRPNT           ;GO TO READ CHECK ERROR PRINT
        IN       CDATA            ;CLEAR KEYBOARD
        IN       0FFH             ;READ SENSE SWITCHES
        ANI      01H              ;SWITCH 0
        JNZ      CONT
        LXI      H,MSG8
        CALL     PMSG             ;REQUEST INPUT
        CALL     CONIN            ;READ KEYS
        CPI      'R'              ;CHECK FOR R
        JZ       FIX
        CPI      'C'              ;CHECK FOR C
        JZ       CONT
        HLT
FIX:    MVI      A,01             ;SET REPETE FLAG
        STA      REPETE
        CALL     CRLF
```

```
         CALL     CRLF
         RET
CONT:    CALL     CRLF
         CALL     CRLF
         RET
;
DATERR:  STA      BADBT            ;SAVE BAD BYTE
         LDA      ERRFLG           ;LOAD ERROR COUNT
         INR      A
         STA      ERRFLG           ;NEW COUNT
         JMP      ERRET            ;RETURN
;
;
; WRITE THE SECTOR AT SECT, ON THE PRESENT TRACK.
; USE STARTING ADDRESS AT DMAADD.
;  LOAD HEAD FIRST
;
WRITE:   LDA       PATEN
         MOV       B,A     ;TEST PATTERN IN B
         LDA       SECT    ;LOAD SECTOR
WRITE1:  OUT   SECTP       ;SET THE SECTOR INTO 1771.
         MVI   A,0ACH      ;SET UP 1771 FOR WRITE.
         OUT   DCOM
WLOOP:   IN    WAIT        ;WAIT FOR READY.
         ORA   A           ;SET FLAGS.
         JP    WDONE       ;HOP OUT WHEN DONE.
;
;     INSERT PATTERN HERE
         MOV       A,B     ;LOAD TEST PATTERN
;
;
         OUT   DDATA       ;WRITE ONTO DISK.
         INX   H           ;INCREMENT MEM PTR.
         JMP   WLOOP       ;KEEP WRITING.
WDONE:   IN    DSTAT       ;READ DISK STATUS.
         ANI   0FDH        ;LOOK AT THESE BITS.
         MOV       D,A     ;SAVE STATUS BITS
PROCER:  RZ                ;RETURN IF NO ERR.
WERRO:   LXI   H,WTMSG     ;PRINT "WRITE ".
         CALL  PMSG
         MOV   A,D         ;GET ERROR BITS.
         ANI   40H         ;LOOK AT BIT 6.
         LXI   H,WPMSG     ;PRINT "PROTECT ".
         CNZ   PMSG
         MOV   A,D         ;GET ERROR BITS.
         ANI   20H         ;LOOK AT BIT 5.
         LXI   H,WFMSG     ;PRINT "FAULT ".
         CNZ   PMSG
         JMP   ERMSG1      ;DO COMMON MESSAGES.
;
; MOVE THE HEAD TO THE TRACK IN REGISTER A.
;
SEEK:    OUT   DDATA       ;TRACK TO DATA REGISTER.
BUSY:    IN    DSTAT       ;READ DISK STATUS.
         RRC               ;LOOK AT BIT 0.
         JC    BUSY        ;WAIT TILL NOT BUSY.
         MVI   A,12H       ;SET FOR 10 MS STEP.
         ORI   4           ;VERIFY ON LAST TRACK.
         OUT   DCOM        ;ISSUE SEEK COMMAND.
         IN    WAIT        ;WAIT FOR INTRQ.
         IN    DSTAT       ;READ STATUS.
         ANI   91H         ;LOOK AT BITS.
         MOV       D,A     ; SAVE STATUS
         RZ                ;RETURN IF NO ERROR
         LXI   H,SKMSG     ;PRINT "SEEK ".
         JMP   ERMSG       ;DO COMMON ERR MESSAGES.
;
; PRINT THE MESSAGE AT H&L UNTIL A ZERO.
;
PMSG:    MOV   A,M         ;GET A CHARACTER.
         ORA   A           ;IF IT'S ZERO,
         RZ                ;RETURN.
         MOV   C,A         ;OTHERWISE,
         CALL  CONOT       ;PRINT IT.
         INX   H           ;INCREMENT H&L,
         JMP   PMSG        ;AND GET ANOTHER.
```

```
; CBIOS MESSAGES
RENT    EQU      0000H   ;MONITOR ENTRY
;
;
NRMSG:  DB    'NOT READY ',0
RNMSG:  DB    'RECORD NOT FOUND ',0
CRCMSG: DB    'CRC ',0
LDMSG:  DB    'LOST DATA ',0
BSYMSG: DB    'BUSY ',0
WPMSG:  DB    'PROTECT ',0
WFMSG:  DB    'FAULT ',0
ERRMSG: DB    'ERROR.',0
RDMSG:  DB    ODH,OAH,'READ ',0
WTMSG:  DB    ODH,OAH,'WRITE ',0
SKMSG:  DB    ODH,OAH,'SEEK ',0
HEMSG:  DB    ODH,OAH,'HOME ',0
MNTMSG: DB    ODH,OAH,'MOUNT ',0

;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;    PRINT 8 BIT WORD IN BINARY FORMAT
;       INPUT:  DATA IN REG A
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;
BITS:   MOV     B,A      ; DATA
        MVI     A,80H    ; MASK
OVER:   MVI     C,30H
        MOV     E,A      ; STORE MASK
        ANA     B        ; AND WITH MASK
        JZ      PRNT     ; JUMP IF ZERO
        MVI     C,31H
PRNT:   CALL    CONOT
        ANA     B        ; ZERO CARRY
        MOV     A,E      ; LOAD MASK
        RAR
        JNC     OVER
        RET
;;
;
BLNK:   MVI     C,20H             ;PRINT BLANKS, # IN REG. D
LP1:    CALL    CONOT1
        DCR     D
        JNZ     LP1
        RET
;
BINHA:  MOV     A,D
        RAR
        RAR
        RAR
        RAR
        CALL    BIN1
        MOV     C,A
        CALL    CONOT
        MOV     A,D
        CALL    BIN1
        MOV     C,A
        CALL    CONOT
        RET
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;    OUTPUTS FOUR HEX DIGITS IN ASCII
;    ENTER WITH DATA IN REG PAIR E AND D
;;;;;;;;;;;;;;;;;;;
;;
;
BINB:   CALL    BINHA
        MOV     D,E
        CALL    BINHA
        RET
```

```
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;     CONVERTS HEX TO ASCII
;     INPUT: 4 BITS HEX  REG A
;     OUTPUT:  8 BIT ASSCII REG A
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;
;
BIN1:    ANI      0FH
         ADI      30H
         CPI      3AH
         RC
         ADI      07H
         RET
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;      INITIATE SIO PORTS
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;
;
INITA:   MVI      A,0AAH              ;GET DUMMY MODE WORD
          OUT     CSTAT               ;OUTPUT IT
         MVI      A,40H               ;GET RESET BIT
         OUT      CSTAT               ;RESET SIO BOARD
         MVI      A,0CEH              ;GET REAL MODE WORD
         OUT      CSTAT               ;SET THE MODE FOR REAL
         MVI      A,37H               ;GET THE COMMAND
         OUT      CSTAT               ;OUTPUT IT
         RET
;
;
;
CRLF:    MVI      C,13                ;CR
         CALL     CONOT
LF:      MVI      C,10                ;LF
         CALL     CONOT1
         MVI      C,7FH
         CALL     CONOT1
         CALL     CONOT
         RET
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;
;
;
         END
```

Section 8

Track Write Routine

8.1 General Description

The track write routine is not a diagnostic test, but rather a programmed aid to be used during disk drive maintenance and alignment. When executed, an entire track, as selected in the sense switches, is repeatedly written with an all ones pattern.

8.2 Program Details

The disk track write routine does not test the data written, but it does test and report on the ability of the disk drive to load the read/write head and to move it to a selected track.

The track write routine reads the IMSAI 8080 front panel sense switches for the desired track. Error checking is performed to test if the selected track is greater than 76, the last track. If the selected track number is greater than 76, the front panel sense lites oscillate and no writing is performed.

If the selected track number is valid, a seek to that track is performed. The read/write head is then loaded and full track writing can begin at the next index marker. The entire track is written with all ones until the index mark is reached again. The track is repeatedly written until a new track is selected by the operator.

This continuous writing can be very valuable when troubleshooting or aligning the disk drive heads and read/write circuitry as outlined in the Pertec disk drive manual.

8.3 Operation

Upon entering the program, the routine will request that a scratch disk be mounted. The scratch disk does not need to be formatted for this routine. After the scratch disk is loaded, the desired track should be set in the sense switches and then type a 'Y' on the console to begin. The drive will then perform a seek to that track and begin writing.

The track number may be changed at any time and the sequence will begin on the new track. A convenient way to stop the writing at any time is to raise the left most sense switch which halts the write operations by forcing a large track number.

Some drive error messages ask the operator whether to abort or continue after the error.

The track write routine is stored in programmable memory on the diagnostic memory board. The write routine can be run under the diagnostic operating system test controller, or it can be started from the IMSAI 8080 front panel at a starting address of 'CD80' Hex.

The track write routine is also stored on floppy disk and it can be invoked by the CPM operating system.

under file name 'WRTTRK.COM'.

```
; DISC TEST FULL TRACK WRITE
;    SELECT TRACK IN SENSE SWITCHES
;    DISC VERSION
;    BRIAN CONLAN
;    JUNE 79
;
         ORG      0100H
;

ENTRYA: JMP      STARTA
ENTRYB: LXI      H,080H
        SPHL
        DI                          ;SET STACK
        CALL     INITA
STARTA: LXI      H,MSG1B            ;RESET SIO
READT:  CALL     PMSG
        CALL     CONIN
        CPI      'Y'                ;READ KEYBOARD
        LXI      H,MSG2A
        JNZ      READT
        CALL     HOME
STARTB: ORA      A
        JNZ      STARTA             ;SET FLAGS
        LXI      H,STARTB           ;ERROR START OVER
        PUSH     H                  ;SUBROUTINE RETURN
STARTC: IN       OFFH
        CPI      77                 ;READ SENSE SWITCHES
        JNC      ERRA               ;PREVENT TRACK OVER-DRIVE
SEEKA:  CALL     SEEK
        MVI      B,OFFH             ;MOVE HEAD TO TRACK
        MVI      A,OF4H             ;TEST PATTERN
        OUT      DCOM               ;WRITE TRK COMMAND
WRTLP:  IN       WAIT
        ORA      A
        JP       WDONE
        MOV      A,B
        OUT      DDATA
        JMP      WRTLP
;
;
;
ERRA:   MVI      B,OFOH
ERRLP:  MOV      A,B
        CMA
        OUT      OFFH
        LXI      D,01H
        LXI      H,000H
ERRLPB: DAD      D                  ;DELAY LOOP
        JNC      ERRLPB
        MOV      B,A
        IN       OFFH
        CPI      77                 ;SEE IF SWITCHES FIXED
        JNC      ERRLP
DELAY:  LXI      H,0
        LXI      D,01H
DELP:   DAD      D
        JNC      DELP
DELAYA: LXI      H,0
        LXI      D,01H
DELPA:  DAD      D
        JNC      DELPA
        JMP      STARTC
```

```
MSG1B:  DB      0DH,0AH,0AH,'DISK TRACK WRITE ROUTINE'
        DB      0DH,0AH,'LOAD SCRATCH DISK  TYPE Y WHEN READY ',0
MSG2A:  DB      0DH,0AH,'?? ',0
;
;
;
LPCNT:  DS      1               ;SPACE FOR LOOP COUNTER
BADBT:  DS      1               ;SPACE FOR BAD BYTE
BDTRK:  DS      1               ;SPACE FOR DISK READ TRK WHEN ERR
MSG1:   DB      0DH,0AH,'DISK TEST NO. 1 FORMATTED TEST  ',0
MSG1A:  DB      0DH,0AH,'LOAD SCRATCH DISK  TYPE Y WHEN READY',0
MSG2    DB      0DH,0AH,' END OF PASS ',0
MSG3:   DB      0DH,0AH,'DATA ERROR ON DISK CHECK       ERROR COUNT IN HEX   ',0
MSG4:   DB      0DH,0AH,' TRACK NO.        SECTOR NO.       TEST BYTE         LAST ERROR',0DH,
MSG5:   DB      0DH,0AH,'HEAD POSITION ',0
MSG6:   DB      0DH,0AH,'DISK TRACK           CONTROLLER TRACK          SECTOR ',0DH,0AH,0
MSG7:   DB      0DH,0AH,0DH,0AH,'  !! EXECUTION STOPPED !! ',0
MSG8:   DB      0DH,0AH,'TYPE R TO RETRY,  C TO CONTINUE,  ANYTHING ELSE STOP ',0
;
CSTAT   EQU  3          ;CONSOLE STATUS PORT.
CCOM    EQU  3          ;CONSOLE COMMAND PORT.
CDATA   EQU  2          ;CONSOLE DATA PORT.
CKBR    EQU  00000010B  ;KEYBOARD READY BIT.
CPTR    EQU  00000001B  ;PRINT READY BIT.
CNULL   EQU  1          ;CONSOLE NULL COUNT.
DISK    EQU  0FBH       ;DISK BASE ADDRESS.
DCOM    EQU  DISK       ;DISK COMMAND PORT.
DSTAT   EQU  DISK       ;DISK STATUS PORT.
TRACK   EQU  DISK+1     ;DISK TRACK PORT.
SECTP   EQU  DISK+2     ;DISK SECTOR PORT.
DDATA   EQU  DISK+3     ;DISK DATA PORT.
WAIT    EQU  DISK+4     ;DISK WAIT PORT.
DCONT   EQU  DISK+4     ;DISK CONTROL PORT.

TRK:    DS      1               ;ADDRESS FOR TRACK
SECT:   DS      1               ;ADDRESS FOR SECTOR

;
; READ A CHARACTER FROM CONSOLE.
;
CONIN:   IN   CSTAT             ;READ CONSOLE STATUS.
         ANI  CKBR      ;IF NOT READY,

        JZ   CONIN      ;READY WHEN HIGH.
         IN   CDATA     ;READ A CHARACTER.
        OUT     CDATA
         ANI  7FH       ;MAKE MOST SIG. BIT = 0.
         RET
;
; WRITE A CHARACTER TO THE CONSOLE DEVICE.
;
CONOT:  MVI  A,0DH      ;IF IT'S A CR,
        CMP  C          ;THEN HOP OUT
        JZ   CONUL      ;TO NULL ROUTINE.
CONOT1: IN   CSTAT      ;READ CONSOLE STATUS.
        ANI  CPTR       ;IF NOT READY,
        JZ   CONOT1     ;READY WHEN HIGH.
        MOV  A,C        ;GET CHARACTER.
        OUT  CDATA      ;PRINT IT.
        RET             ;RETURN.
CONUL:  PUSH B          ;SAVE B&C.
        MVI  B,CNULL    ;GET NULL COUNT.
CONUL1: CALL CONOT1     ;PRINT CR.
        MVI  C,0        ;GET NULL CHAR.
        DCR  B          ;DECREMENT COUNTER.
        JNZ  CONUL1     ;DO NEXT NULL.
        POP  B          ;RESTORE B&C.
        MOV  A,C        ;RESTORE A.
        RET             ;RETURN.
```

```
; MOVE DISK TO TRACK ZERO.
;
HOME:    MVI  A,0D0H        ;CLEAR ANY PENDING COMMAND.
         OUT  DCOM
         XRA     A          ;ZERO ACC
         STA      TRK       ;STORE TRACK
HOME1:   IN   DSTAT         ;READ DISK STATUS.
         RRC                ;LOOK AT LSB.
         JC   HOME1         ;WAIT FOR NOT BUSY.
         MVI  A,3           ;20 MS STEP RATE.
         OUT  DCOM          ;ISSUE HOME COMMAND.
         IN   WAIT          ;WAIT FOR INTRQ.
         ORA  A             ;SET FLAGS.
         JM   HERR          ;ERROR IF DRQ.
         IN   DSTAT         ;READ DISK STATUS.
         MOV  D,A           ;SAVE IN REGISTER D.
         ANI  4             ;LOOK AT BIT 2.
         JZ   HERR          ;ERROR IF NOT TRK 0.
         MOV  A,D           ;GET STATUS BACK.
         ANI  91H           ;MASK NON-ERROR BITS.
         RZ                 ;RETURN IF NO ERROR.
HERR:    LXI  H,HEMSG       ;PRINT "HOME ".
         MOV  A,D           ;MASK NON-ERROR BITS.
         ANI  91H
         MOV  D,A
         JMP  ERMSG          ;DO COMMON ERROR MSGS.
;
;
ERMSG:   CALL PMSG           ;PRINT ORIGIN MESSAGE.
;
;
;        COMMON ERROR PRINT OUT
;

ERMSG1: MOV  A,D            ;GET ERROR BITS.
        ANI  80H            ;IF BIT 7 HIGH,
        LXI  H,NRMSG        ;"NOT READY".
        CNZ PMSG
        MOV  A,D            ;GET ERROR BITS.
        ANI  10H            ;IF BIT 4 IS HIGH,
        LXI  H,RNMSG        ;PRINT "RECORD NOT FOUND"
        CNZ PMSG
        MOV  A,D            ;GET ERROR BITS.
        ANI  8H             ;IF BIT 3 IS HIGH,
        LXI  H,CRCMSG       ;PRINT 'CRC ERROR".
        CNZ PMSG
        MOV  A,D            ;GET ERROR BITS.
        ANI  4H             ;IF BIT 2 IS HIGH,
        LXI  H,LDMSG        ;PRINT "LOST DATA".
        CNZ PMSG
        MOV  A,D            ;GET ERROR BITS.
        ANI  1              ;IF BIT 1 IS HIGH,
        LXI  H,BSYMSG       ;PRINT "BUSY".
        CNZ  PMSG
PERMSG: LXI  H,ERRMSG       ;PRINT "ERROR."
        CALL PMSG
        MOV      A,D                ;MOVE FLAGS TO ACC
        ANI      18H                ;CRC OR RECORD NOT FOUND
        JZ       RETRY
TRKCHK: MVI      A,0C4H
        OUT      DCOM               ;READ ADDRESS
        IN       WAIT
        IN       DDATA              ;TRACK ADDRESS
        STA      BDTRK
CHKS2   IN       WAIT               ;DUMP REST OF DATA
        JM       CHKS2
        LXI      H,MSG5             ;HEAD ERROR MESSAGE
        CALL     PMSG
        LXI      H,MSG6             ;HEADINGS
        CALL     PMSG
        MVI      D,05H
        CALL     BLNK               ;SPACE OVER
        LDA     BDTRK           ;DISK TRK
        MOV     D,A
        CALL    BINHA           ;PRINT TRK
        MVI     D,15H
        CALL    BLNK            ;SPACE OVER
        IN      TRACK
        MOV     D,A
        CALL    BINHA           ;PRINT TRK
        MVI     D,13H
        CALL    BLNK
        LDA     SECT            ;SECTOR
        MOV     D,A
        CALL    BINHA           ;PRINT SECTO NO.
RETRY;  IN      CDATA           ;CLEAR KEYBOARD
        IN      OFFH            ;READ SENSE SWITCHES
        ANI     080H            ;SWITCH 0
        JNZ     CONT
        LXI     H,MSG8
        CALL    PMSG            ;REQUEST INPUT
        CALL    CONIN           ;READ KEYS
        CPI     'R'             ;CHECK FOR R
        JZ      FIX
        CPI     'C'             ;CHECK FOR C
        JZ      CONT
        HLT
FIX:    MVI     A,01            ;SET REPETE FLAG
        CALL    CRLF
        CALL    CRLF
        RET
CONT:   CALL    CRLF
        CALL    CRLF
        RET
;
;
;
WDONE:  IN   DSTAT          ;READ DISK STATUS.
        ANI  OFDH           ;LOOK AT THESE BITS.
        MOV     D,A         ;SAVE STATUS BITS
PROCER: RZ                  ;RETURN IF NO ERR.
WERRO:  LXI  H,WTMSG        ;PRINT "WRITE ".
        CALL PMSG
        MOV  A,D            ;GET ERROR BITS.
        ANI  40H            ;LOOK AT BIT 6.
        LXI  H,WPMSG        ;PRINT "PROTECT ".
        CNZ  PMSG
        MOV  A,D            ;GET ERROR BITS.
        ANI  20H            ;LOOK AT BIT 5.
        LXI  H,WFMSG        ;PRINT "FAULT ".
        CNZ  PMSG
        JMP  ERMSG          ;DO COMMON MESSAGES.
;
; MOVE THE HEAD TO THE TRACK IN REGISTER A.
;
SEEK:   OUT  DDATA          ;TRACK TO DATA REGISTER.
BUSY:   IN   DSTAT          ;READ DISK STATUS.
        RRC                 ;LOOK AT BIT 0.
        JC   BUSY           ;WAIT TILL NOT BUSY.
        MVI  A,12H          ;SET FOR 10 MS STEP.
        OUT  DCOM           ;ISSUE SEEK COMMAND.
        IN   WAIT           ;WAIT FOR INTRQ.
        IN   DSTAT          ;READ STATUS.
        ANI  91H            ;LOOK AT BITS.
        MOV     D,A         ; SAVE STATUS
        RZ                  ;RETURN IF NO ERROR
        LXI  H,SKMSG        ;PRINT "SEEK ".
        JMP  ERMSG          ;DO COMMON ERR MESSAGES.
;
; PRINT THE MESSAGE AT H&L UNTIL A ZERO.
;
PMSG:   MOV  A,M            ;GET A CHARACTER.
        ORA  A              ;IF IT'S ZERO,
        RZ                  ;RETURN.
        MOV  C,A            ;OTHERWISE,
        CALL CONOT          ;PRINT IT.
        INX  H              ;INCREMENT H&L,
        JMP  PMSG           ;AND GET ANOTHER.
```

```
;
; CBIOS MESSAGES
RENT    EQU     0000H   ;MONITOR ENTRY
;
;
NRMSG:  DB   'NOT READY ',0
RNMSG:  DB   'RECORD NOT FOUND ',0
CRCMSG: DB   'CRC ',0
LDMSG:  DB   'LOST DATA ',0
BSYMSG: DB   'BUSY ',0
WPMSG:  DB   'PROTECT ',0
WFMSG:  DB   'FAULT ',0
ERRMSG: DB   'ERROR.',0
RDMSG:  DB   ODH,OAH,'READ ',0
WTMSG:  DB   ODH,OAH,'WRITE ',0
SKMSG:  DB   ODH,OAH,'SEEK ',0
HEMSG:  DB   ODH,OAH,'HOME ',0
MNTMSG: DB   ODH,OAH,'MOUNT ',0

;;;;;;;;;;;;;;;;;;;;;;
;
BLNK:   MVI     C,20H           ;PRINT BLANKS, # IN REG. D
LP1:    CALL    CONOT1
        DCR     D
        JNZ     LP1
        RET
;
BINHA:  MOV     A,D
        RAR
        RAR
        RAR
        RAR
        CALL    BIN1
        MOV     C,A
        CALL    CONOT
        MOV     A,D
        CALL    BIN1
        MOV     C,A
        CALL    CONOT
        RET
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;    OUTPUTS FOUR HEX DIGITS IN ASCII
;    ENTER WITH DATA IN REG PAIR E AND D
;;;;;;;;;;;;;;;;;;
;;
;
BINB:   CALL    BINHA
        MOV     D,E
        CALL    BINHA
        RET
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;    CONVERTS HEX TO ASCII
;    INPUT: 4 BITS HEX  REG A
;    OUTPUT:  8 BIT ASSCII REG A
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;
;
BIN1:   ANI     OFH
        ADI     30H
        CPI     3AH
        RC
        ADI     07H
        RET
```

```
,
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;     INITIATE SIO PORTS
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;
;
INITA:  MVI     A,0AAH          ;GET DUMMY MODE WORD
         OUT    CSTAT           ;OUTPUT IT
        MVI     A,40H           ;GET RESET BIT
        OUT     CSTAT           ;RESET SIO BOARD
        MVI     A,0CEH          ;GET REAL MODE WORD
        OUT     CSTAT           ;SET THE MODE FOR REAL
        MVI     A,37H           ;GET THE COMMAND
        OUT     CSTAT           ;OUTPUT IT
        RET
;
;
;
CRLF:   MVI     C,13            ;CR
        CALL    CONOT
LF:     MVI     C,10            ;LF
        CALL    CONOT1
        MVI     C,7FH
        CALL    CONOT1
        CALL    CONOT
        RET
```

Section 9
Track Read Routine

9.1 General Description

The Track Read Routine is not a diagnostic test, but rather a programmed aid to be used during disk drive maintenance and alignment. When executed, an entire track as selected on the sense switches, is repeatedly read.

9.2 Program Details

The disk drive read routine does not save or test the data being read, but it does test and report on the ability of the disk drive to load the read/write head and to move it to a selected track.

The track read routine reads the IMSAI 8080 front panel sense switches for the desired track. Error checking is performed to test if the selected track number is greater than 76, the last track. If the selected number is greater than 76, the front panel sense lites oscillate and no reading is performed.

If the track number is valid, a seek to that track is performed. The read/write head is then loaded so full track reading can begin on the next index marker. The entire track is read, but the data is not saved. The track is repeatedly read until a new track is selected by the operator.

This continous reading can be very valuable when trouble shooting or aligning the disk head or read/write circuitry as outlined in the Pertec disk drive manual.

9.3 Operation

Upon entering the program, the routine will request that a scratch disk be mounted. The scratch disk does not need to be formatted for this routine. After a scratch disk is loaded, the desired track should be set in the sense switches a . then type a 'Y' on the console to begin. The drive will then perform a seek to that track and begin reading.

The track number may be changed at anytime and the sequence will begin on the new track. A convenient way to stop the reading at any time is to raise the left most sense switch which halts the read operations by forcing a large track number.

Some drive error asks the operator whether to abort or continue after the error.

The track read routine is stored in programmable memory on the diagnostic board. The read routine can be run under the diagnostic operating system test controller, or it can be started from the IMSAI 8080 front panel at a starting address of 'CE40'.

The read routine is also stored on floppy disk and it can be invoked by the CPM operating system under file name 'RDTRK.COM'.

```
;
; DISC TEST FULL TRACK READ
;   SELECT TRACK IN SENSE SWITCHES
;    DISC VERSION
;    BRIAN DONLAN
;    JUNE 79
;
         ORG      0100H
;

ENTRYA: JMP      STARTA
ENTRYB: LXI      H,080H
        SPHL                        ;SET STACK
        DI
        CALL     INITA              ;RESET SIO
STARTA: LXI      H,MSG1B
READT:  CALL     PMSG
        CALL     CONIN              ;READ KEYBOARD
        CPI      'Y'
        LXI      H,MSG2A
        JNZ      READT
        CALL     HOME
STARTB: ORA      A                  ;SET FLAGS
        JNZ      STARTA             ;ERROR START OVER
        LXI      H,STARTB           ;SUBROUTINE RETURN
        PUSH     H
STARTC: IN       OFFH               ;READ SENSE SWITCHES
        CPI      77                 ;PREVENT TRACK OVER-DRIVE
        JNC      ERRA
SEEKA:  CALL     SEEK               ;MOVE HEAD TO TRACK
        MVI      A,OE5H             ;READ TRACK COMMAND
        OUT      DCOM
RDLP:   IN       WAIT
        ORA      A
        JP       RDONE
        IN       DDATA
        JMP      RDLP
;
;
;
ERRA:   MVI      B,OFOH
ERRLP:  MOV      A,B
        CMA
        OUT      OFFH
        LXI      D,01H
        LXI      H,000H             ;DELAY LOOP
ERRLPB: DAD      D
        JNC      ERRLPB
        MOV      B,A
        IN       OFFH               ;SEE IF SWITCHES FIXED
        CPI      77
        JNC      ERRLP
DELAY:  LXI      H,0
        LXI      D,01H
DELP:   DAD      D
        JNC      DELP
DELAYA· LXI      H,0
        LXI      D,01H
DELPA:  DAD      D
        JNC      DELPA
        JMP      STARTC
```

```
;
RDONE:  IN      DSTAT           ;READ STATUS
        ANI     9DH
        MOV     D,A
        RZ
        LXI     H,RDMSG         ;PRINT 'READ'
        JMP     ERMSG
;
;;
MSG1B:  DB      ODH,OAH,OAH,'DISK TRACK READ ROUTINE'
        DB      ODH,OAH,'LOAD SCRATCH DISK  TYPE Y WHEN READY ',0
MSG2A:  DB      ODH,OAH,'?? ',0
;
;
;
LPCNT:  DS      1               ;SPACE FOR LOOP COUNTER
BADBT:  DS      1               ;SPACE FOR BAD BYTE
BDTRK:  DS      1               ;SPACE FOR DISK READ TRK WHEN ERR
MSG1:   DB      ODH,OAH,'DISK TEST NO. 1 FORMATTED TEST  ',0
MSG1A:  DB      ODH,OAH,'LOAD SCRATCH DISK  TYPE Y WHEN READY',0
MSG2    DB      ODH,OAH,' END OF PASS ',0
MSG3:   DB      ODH,OAH,'DATA ERROR ON DISK CHECK      ERROR COUNT IN HEX   ',0
MSG4:   DB      ODH,OAH,' TRACK NO.       SECTOR NO.      TEST BYTE        LAST ERROR',ODH
MSG5:   DB      ODH,OAH,'HEAD POSITION ',0
MSG6:   DB      ODH,OAH,'DISK TRACK          CONTROLLER TRACK         SECTOR ',ODH,OAH,0
MSG7:   DB      ODH,OAH,ODH,OAH,'  !! EXECUTION STOPPED !! ',0
MSG8:   DB      ODH,OAH,'TYPE R TO RETRY,  C TO CONTINUE,  ANYTHING ELSE STOP ',0
;
CSTAT   EQU  3          ;CONSOLE STATUS PORT.
CCOM    EQU  3          ;CONSOLE COMMAND PORT.
CDATA   EQU  2          ;CONSOLE DATA PORT.
CKBR    EQU  00000010B  ;KEYBOARD READY BIT.
CPTR    EQU  00000001B  ;PRINT READY BIT.
CNULL   EQU  1          ;CONSOLE NULL COUNT.
DISK    EQU  OF8H       ;DISK BASE ADDRESS.
DCOM    EQU  DISK       ;DISK COMMAND PORT.
DSTAT   EQU  DISK       ;DISK STATUS PORT.
TRACK   EQU  DISK+1     ;DISK TRACK PORT.
SECTP   EQU  DISK+2     ;DISK SECTOR PORT.
DDATA   EQU  DISK+3     ;DISK DATA PORT.
WAIT    EQU  DISK+4     ;DISK WAIT PORT.
DCONT   EQU  DISK+4     ;DISK CONTROL PORT.

TRK:    DS      1               ;ADDRESS FOR TRACK
SECT:   DS      1               ;ADDRESS FOR SECTOR

;
; READ A CHARACTER FROM CONSOLE.
;
CONIN:   IN   CSTAT             ;READ CONSOLE STATUS.
         ANI  CKBR      ;IF NOT READY,

        JZ   CONIN      ;READY WHEN HIGH.
         IN   CDATA     ;READ A CHARACTER.
        OUT     CDATA
         ANI  7FH       ;MAKE MOST SIG. BIT = 0.
         RET
;
; WRITE A CHARACTER TO THE CONSOLE DEVICE.
;
CONOT:  MVI  A,ODH      ;IF IT'S A CR,
        CMP  C          ;THEN HOP OUT
        JZ   CONUL      ;TO NULL ROUTINE.
CONOT1: IN   CSTAT      ;READ CONSOLE STATUS.
        ANI  CPTR       ;IF NOT READY,
        JZ   CONOT1     ;READY WHEN HIGH.
        MOV  A,C        ;GET CHARACTER.
        OUT  CDATA      ;PRINT IT.
        RET             ;RETURN.
CONUL:  PUSH B          ;SAVE B&C.
        MVI  B,CNULL    ;GET NULL COUNT.
CONUL1: CALL CONOT1     ;PRINT CR.
        MVI  C,0        ;GET NULL CHAR.
        DCR  B          ;DECREMENT COUNTER.
        JNZ  CONUL1     ;DO NEXT NULL.
        POP  B          ;RESTORE B&C.
        MOV  A,C        ;RESTORE A.
        RET             ;RETURN.
```

```
; MOVE DISK TO TRACK ZERO.
;
HOME:     MVI  A,0D0H        ;CLEAR ANY PENDING COMMAND.
          OUT  DCOM
          XRA     A          ;ZERO ACC
          STA     TRK        ;STORE TRACK
HOME1:    IN   DSTAT         ;READ DISK STATUS.
          RRC                ;LOOK AT LSB.
          JC   HOME1         ;WAIT FOR NOT BUSY.
          MVI  A,3           ;20 MS STEP RATE.
          OUT  DCOM          ;ISSUE HOME COMMAND.
          IN   WAIT          ;WAIT FOR INTRQ.
          ORA  A             ;SET FLAGS.
          JM   HERR          ;ERROR IF DRQ.
          IN   DSTAT         ;READ DISK STATUS.
          MOV  D,A           ;SAVE IN REGISTER D.
          ANI  4             ;LOOK AT BIT 2.
          JZ   HERR          ;ERROR IF NOT TRK 0.
          MOV  A,D           ;GET STATUS BACK.
          ANI  91H           ;MASK NON-ERROR BITS.
          RZ                 ;RETURN IF NO ERROR.
HERR:     LXI  H,HEMSG       ;PRINT "HOME ".
          MOV  A,D           ;MASK NON-ERROR BITS.
          ANI  91H
          MOV  D,A
          JMP  ERMSG         ;DO COMMON ERROR MSGS.
;
;
ERMSG:    CALL PMSG          ;PRINT ORIGIN MESSAGE.
;
;
;         COMMON ERROR PRINT OUT
;

ERMSG1:   MOV  A,D           ;GET ERROR BITS.
          ANI  80H           ;IF BIT 7 HIGH,
          LXI  H,NRMSG       ;"NOT READY".
          CNZ  PMSG
          MOV  A,D           ;GET ERROR BITS.
          ANI  10H           ;IF BIT 4 IS HIGH,
          LXI  H,RNMSG       ;PRINT "RECORD NOT FOUND"
          CNZ  PMSG
          MOV  A,D           ;GET ERROR BITS.
          ANI  8H            ;IF BIT 3 IS HIGH,
          LXI  H,CRCMSG      ;PRINT "CRC ERROR".
          CNZ  PMSG
          MOV  A,D           ;GET ERROR BITS.
          ANI  4H            ;IF BIT 2 IS HIGH,
          LXI  H,LDMSG       ;PRINT "LOST DATA".
          CNZ  PMSG
          MOV  A,D           ;GET ERROR BITS.
          ANI  1             ;IF BIT 1 IS HIGH,
          LXI  H,BSYMSG      ;PRINT "BUSY".
          CNZ  PMSG
PERMSG:   LXI  H,ERRMSG      ;PRINT "ERROR."
          CALL PMSG
          MOV     A,D                ;MOVE FLAGS TO ACC
          ANI     18H                ;CRC OR RECORD NOT FOUND
          JZ      RETRY
TRKCHK:   MVI     A,0C4H
          OUT     DCOM               ;READ ADDRESS
          IN      WAIT
          IN      DDATA              ;TRACK ADDRESS
          STA     BDTRK
CHKS2     IN      WAIT               ;DUMP REST OF DATA
          JM      CHKS2
          LXI     H,MSG5             ;HEAD ERROR MESSAGE
          CALL    PMSG
          LXI     H,MSG6             ;HEADINGS
          CALL    PMSG
          MVI     D,05H
          CALL    BLNK               ;SPACE OVER
          LDA     BDTRK              ;DISK TRK
          MOV      D,A
          CALL     BINHA            ;PRINT TRK
          MVI      D,15H
          CALL     BLNK             ;SPACE OVER
          IN       TRACK
          MOV      D,A
          CALL     BINHA            ;PRINT TRK
          MVI      D,13H
          CALL     BLNK
          LDA      SECT             ;SECTOR
          MOV      D,A
          CALL     BINHA            ;PRINT SECTO NO.
 RETRY;   IN       CDATA            ;CLEAR KEYBOARD
          IN       OFFH             ;READ SENSE SWITCHES
          ANI      080H             ;SWITCH 0
          JNZ      CONT
          LXI      H,MSG8
          CALL     PMSG             ;REQUEST INPUT
          CALL     CONIN            ;READ KEYS
          CPI      'R'              ;CHECK FOR R
          JZ       FIX
          CPI      'C'              ;CHECK FOR C
          JZ       CONT
          HLT
FIX:      MVI      A,01             ;SET REPETE FLAG
          CALL     CRLF
          CALL     CRLF
          RET
CONT:     CALL     CRLF
          CALL     CRLF
          RET
;
;
;
WDONE:    IN    DSTAT       ;READ DISK STATUS.
          ANI   OFDH        ;LOOK AT THESE BITS.
          MOV      D,A      ;SAVE STATUS BITS
PROCER:   RZ                ;RETURN IF NO ERR.
WERRO:    LXI   H,WTMSG     ;PRINT "WRITE ".
          CALL  PMSG
          MOV   A,D         ;GET ERROR BITS.
          ANI   40H         ;LOOK AT BIT 6.
          LXI   H,WPMSG     ;PRINT "PROTECT ".
          CNZ   PMSG
          MOV   A,D         ;GET ERROR BITS.
          ANI   20H         ;LOOK AT BIT 5.
          LXI   H,WFMSG     ;PRINT "FAULT ".
          CNZ   PMSG
          JMP   ERMSG1      ;DO COMMON MESSAGES.
;
; MOVE THE HEAD TO THE TRACK IN REGISTER A.
;
SEEK:     OUT   DDATA       ;TRACK TO DATA REGISTER.
BUSY:     IN    DSTAT       ;READ DISK STATUS.
          RRC               ;LOOK AT BIT 0.
          JC    BUSY        ;WAIT TILL NOT BUSY.
          MVI   A,12H       ;SET FOR 10 MS STEP.
          OUT   DCOM        ;ISSUE SEEK COMMAND.
          IN    WAIT        ;WAIT FOR INTRQ.
          IN    DSTAT       ;READ STATUS.
          ANI   91H         ;LOOK AT BITS.
          MOV      D,A      ; SAVE STATUS
          RZ                ;RETURN IF NO ERROR
          LXI   H,SKMSG     ;PRINT "SEEK ".
          JMP   ERMSG       ;DO COMMON ERR MESSAGES.
;
; PRINT THE MESSAGE AT H&L UNTIL A ZERO.
;
PMSG:     MOV   A,M         ;GET A CHARACTER.
          ORA   A           ;IF IT'S ZERO,
          RZ                ;RETURN.
          MOV   C,A         ;OTHERWISE,
          CALL  CONOT       ;PRINT IT.
          INX   H           ;INCREMENT H&L,
          JMP   PMSG        ;AND GET ANOTHER.
```

```
;
; CBIOS MESSAGES
RENT     EQU       0000H    ;MONITOR ENTRY
;
;
NRMSG:  DB    'NOT READY ',0
RNMSG:  DB    'RECORD NOT FOUND ',0
CRCMSG: DB    'CRC ',0
LDMSG:  DB    'LOST DATA ',0
BSYMSG: DB    'BUSY ',0
WPMSG:  DB    'PROTECT ',0
WFMSG:  DB    'FAULT ',0
ERRMSG: DB    'ERROR.',0
RDMSG:  DB    ODH,OAH,'READ ',0
WTMSG:  DB    ODH,OAH,'WRITE ',0
SKMSG:  DB    ODH,OAH,'SEEK ',0
HEMSG:  DB    ODH,OAH,'HOME ',0
MNTMSG: DB    ODH,OAH,'MOUNT ',0

;;;;;;;;;;;;;;;;;;;;;
;
BLNK:   MVI     C,20H              ;PRINT BLANKS, # IN REG. D
LP1:    CALL    CONOT1
        DCR     D
        JNZ     LP1
        RET
;
BINHA:  MOV     A,D
        RAR
        RAR
        RAR
        RAR
        CALL    BIN1
        MOV     C,A
        CALL    CONOT
        MOV     A,D
        CALL    BIN1
        MOV     C,A
        CALL    CONOT
        RET
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;    OUTPUTS FOUR HEX DIGITS IN ASCII
;    ENTER WITH DATA IN REG PAIR E AND D
;;;;;;;;;;;;;;;;;
;;
;
BINB:   CALL    BINHA
        MOV     D,E
        CALL    BINHA
        RET
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;    CONVERTS HEX TO ASCII
;    INPUT: 4 BITS HEX  REG A
;    OUTPUT:  8 BIT ASSCII REG A
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;
;
BIN1:   ANI     OFH
        ADI     30H
        CPI     3AH
        RC
        ADI     07H
        RET
```

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;      INITIATE SIO PORTS
;
;;;;;;;;  ,;;;;;;;;;;;;;;;;;;;;;;;
;;
;
INITA:  MVI      A,0AAH            ;GET DUMMY MODE WORD
         OUT     CSTAT             ;OUTPUT IT
        MVI      A,40H             ;GET RESET BIT
        OUT      CSTAT             ;RESET SIO BOARD
        MVI      A,0CEH            ;GET REAL MODE WORD
        OUT      CSTAT             ;SET THE MODE FOR REAL
        MVI      A,37H             ;GET THE COMMAND
        OUT      CSTAT             ;OUTPUT IT
        RET
;
;
;
CRLF:   MVI      C,13              ;CR
        CALL     CONOT
LF:     MVI      C,10              ;LF
        CALL     CONOT1
        MVI      C,7FH
        CALL     CONOT1
        CALL     CONOT
        RET
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
```

Section 10

Unibus Port Test

10.1 General Description

The Unibus Port Test checks the S-100 to Unibus adapter and the Unibus data and address lines.

10.2 Program Details

The Unibus is a wired-or bus and any device on the bus can pull a data or address line low (logical 1). When not in use all data and address lines should be high (logical 0). The port test continuously reads the Unibus and in a cyclic fashion, sets a line low and then verifies that this and only this line is low (logical 1). The test then proceeds to the next line and tests it until all data and address lines are tested. The port test uses the Unibus adapter to read and write to the Unibus lines, verifying their operation. If an error is detected, the Unibus adapter port number and the actual and expected bit patterns are printed on the console. Figure 10.1 shows the relationship between the IMSAI adapter port numbers and Unibus data and address lines.

This test has proven particularly helpful in locating problems arising due to misaligned cards in the Unibus card cage where address and data lines become shorted together.

10.3 Operation

The Unibus Port Test is self-contained and requires no operator responses once running. The test checks all the data lines, reports any errors, and returns to the monitor automatically. There are no console or sense switch inputs.

The Unibus Port test is stored in programmable memory on the diagnostic memory board. The program can be run under the diagnostic operating system test contoller, or it can be started from the IMSAI 8080 front panel at a starting address of 'D000' Hex.

The port test is also stored on floppy disk and it can be invoked by the CPM operating system under file name 'UBPORT.COM'.

Unibus Signal									Port No.
HI Address A 16:09	A16	A15	A14	A13	A12	A11	A10	A01	(10)$_{1}$
LO Address A 08:01	A08	A07	A06	A05	A04	A03	A02	A01	(11)$_{1}$
HI Data D 15:08	D15	D14	D13	D12	D11	D10	D09	D08	(12)$_{16}$
LO Data D 07:00	D07	D06	D05	D04	D03	D02	D01	D00	(13)$_{16}$
Control 1	---	---	---	---	---	Ssn	C1	Msn	(14)$_{16}$
Control 2	Int	---	---	---	---	---	---	Grt Req	(15)$_{16}$

Ssn = Slave Sync (input)
C1 = Read/ write (output)
Msn = Master sync (output)
Int = Initiate (output)
Grt = Grant (input)
Req = Request unibus (output)

Fig 10.1 Unibus Adapter Ports

```
         ORG      100H
 ;       UNIBUS PORT TEST
 ;  DISC VERSION  24 MAY 79  B. DONLAN
 ;
 ;
 ENTRY:  LXI      H,ENTRY
         PUSH     H
         DI
         LXI      H,MSG1            ;OPENING MESSAGE
         CALL     PMSG
 ;
 ;BEGINING OF TEST
         MVI      A,01H
         MVI      C,10H             ;PORT UNDER TEST
 PORT10: OUT      10H
         MOV      B,A               ;SAVE TEST PATTERN
         IN       10H               ;READ BUSS
         CMP      B                 ;COMPARE
         CNZ      ERR               ;CALL IF IN ERROR
         RLC
         JNC      PORT10            ;TEST FOR A COMPLETE CICLE
 ;
         MVI      A,01H
         MVI      C,11H             ;PORT 11
 PORT11: OUT      11H
         MOV      B,A               ;SAVE PATTERN
         IN       11H               ;READ BUSS
         CMP      B                 ;C OMPARE
         CNZ      ERR               ;CALL IF ERROR
         RLC
         JNC      PORT11
 ;
 ;
         MVI      A,01H
         MVI      C,12H             ;PORT12
 PORT12: OUT      12H
         MOV      B,A               ;SAVE TEST PATERN
         IN       12H
         CMP      B
         CNZ      ERR
         RLC
         JNC      PORT12
 ;
 ;
         MVI      A,01H
         MVI      C,13H
 PORT13: OUT      13H
         MOV      B,A
         IN       13H
         CMP      B
         CNZ      ERR
         RLC
         JNC      PORT13
 ;
 ;
 ;
 ;
         LXI      H,MSG4            ;FINISHED MESSAGE
         CALL     PMSG
         JMP      RENT              ;RETURN TO MONITOR
```

```
ERR:    PUSH    B                       ;SAVE ERROR PATTERN
        PUSH    A                       ;SAVE TEST PATTERN
        MOV     D,C
        LXI     H,MSG2
        CALL    PMSG
        CALL    BINHA                   ;PRINT 2 DIGITS
        LXI     H,MSG0                  ;ERROR MESSAGE
        CALL    PMSG
        MOV     A,B                     ;LOAD TEST PATTERN
        CALL    BITS                    ;PRINT TEST PATTERN
        LXI     H,MSG3                  ;MORE TEXT
        CALL    PMSG
        POP     A
        CALL    BITS                    ;PRINT ERROR PATTERN
        POP     B                       ;RESTORE B AND C
        MOV     A,B                     ;MOVE TEST PATTERN TO A
        STC
        CMC
        RET
;
;
MSG1    DB      0AH,0AH,0DH,'UNIBUS PORT TEST',0
MSG2:   DB      0AH,0AH,0DH,'ERROR PORT NO.     ',0
MSG0:   DB      0AH,0DH,'TEST PATTERN      ',0
MSG3    DB      0AH,0DH,'ACTUAL PATTERN    ',0
MSG4    DB      0AH,0AH,0DH,'END OF TEST        ',0
RENT    EQU     0000H                   ;MONITOR ENTRY
;
;
;
;
;

; diagnostic input output routines
; for brian donlan  26 feb 79

CSTAT   EQU  3          ;CONSOLE STATUS PORT.
CCOM    EQU  3          ;CONSOLE COMMAND PORT.
CDATA   EQU  2          ;CONSOLE DATA PORT.
CKBR    EQU  00000010B  ;KEYBOARD READY BIT.
CPTR    EQU  00000001B  ;PRINT READY BIT.
CNULL   EQU  1          ;CONSOLE NULL COUNT.

;
; WRITE A CHARACTER TO THE CONSOLE DEVICE.
;
CONOT:  MVI  A,0DH      ;IF IT'S A CR,
        CMP  C          ;THEN HOP OUT
        JZ   CONUL      ;TO NULL ROUTINE.
CONOT1: IN   CSTAT      ;READ CONSOLE STATUS.
        ANI  CPTR       ;IF NOT READY,
        JZ   CONOT1     ;READY WHEN HIGH.
        MOV  A,C        ;GET CHARACTER.
        OUT  CDATA      ;PRINT IT.
        RET             ;RETURN.
CONUL:  PUSH B          ;SAVE B&C.
        MVI  B,CNULL    ;GET NULL COUNT.
CONUL1: CALL CONOT1     ;PRINT CR.
        MVI  C,0        ;GET NULL CHAR.
        DCR  B          ;DECREMENT COUNTER.
        JNZ  CONUL1     ;DO NEXT NULL.
        POP  B          ;RESTORE B&C.
        MOV  A,C        ;RESTORE A.
        RET             ;RETURN.
```

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;        PRINT MESSAGE UNTIL ZERO
;        MESSAGE ADDRESS REG H & L
;
;;;;;;;;;;;;;;;;;;;;
PMSG:   MOV     A,M       ;GET CHAR
        ORA     A         ;IS IT A  ZERO
        RZ
        MOV     C,A       ;OTHERWISE PRINT
        CALL    CONOT
        INX     H         ;INC ADDRESSS
        JMP     PMSG
;
;
;
;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;    PRINT 8 BIT WORD IN BINARY FORMAT
;        INPUT:  DATA IN REG A
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;
BITS:   MOV     B,A       ; DATA
        MVI     A,80H     ; MASK
OVER:   MVI     C,30H
        MOV     E,A       ; STORE MASK
        ANA     B         ; AND WITH MASK
        JZ      PRNT      ; JUMP IF ZERO
        MVI     C,31H
PRNT:   CALL    CONOT
        ANA     B         ; ZERO CARRY
        MOV     A,E       ; LOAD MASK
        RAR
        JNC     OVER
        RET
;;
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;
;     OUTPUTS 2 HEX DIGITS IN ASCCII
;            FROM REG D
;
;;;;;;;;;;;;
;
;
BINHA:  MOV     A,D
        RAR
        RAR
        RAR
        RAR
        CALL    BIN1
        MOV     C,A
        CALL    CONOT
        MOV     A,D
        CALL    BIN1
        MOV     C,A
        CALL    CONOT
        RET
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;    CONVERTS HEX TO ASCII
;    INPUT: 4 BITS HEX  REG A
;    OUTPUT:  8 BIT ASSCII REG A
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;
;
BIN1:   ANI     0FH
        ADI     30H
        CPI     3AH
        RC
        ADI     07H
        RET
```

Section 11

Unibus Communication Test

11.1 General Description

The Unibus Communication Test is one of the most versatile tests written for this system. This test allows an operator to transfer a data word either to or from the console and any device on the Unibus. Error checking is accomplished to monitor the transfer and report any bus errors.

11.2 Program Details

The Unibus Communication Test is completely interactive with the operator responding to the computer requests for data. The test program first requests the transfer mode. There are three valid responses to this request:

I- Input, transfer from Unibus device to console

Control C-exit

O-Output, transfer users data word to Unibus device.

If the output mode is selected, the user is next prompted for the 4 Hex digit data word to be transfered. Error checking is performed on the bus status and operation, but not on the transfered data. A timer is incorporated in the transfer program which will time-out and report on errors if the selected device has made no response after approximately 10 milliseconds. Errors such as bus busy and bus mastership conflicts are also reported in error messages.

11.3 Operation

Since the Unibus Communication Test is interactive, the operator need only respond to the computer's request. All data and address words are 4 hex digits long with no carriage return used.

The communication test is stored in the programmable memory on the diagnostic memory board. The program can be run under the diagnostic operating system controller, or it can be started directly from the IMSAI 8080 front panel at a starting address of 'D100' Hex.

The communication test is also stored on floppy disk and it can be invoked by the CPM operating system under file name 'UBCOMM.COM'.

Unibus Address	Device
FE00	AP-120B Formatter
FE20	AP-120B Word Count
FE21	AP-120B Host Memory Address
FE22	AP-120B DMA Control
FE23	AP-120B AP Memory Address
FE24	AP-120B Panel Switches
FE25	AP-120B Panel Functions
FE26	AP-120B Panel Lites
FE27	AP-120B Reset
FFE8	Filter Control Register
F800	Front Panel
F801	Front Panel
F802	Front Panel
0003	Data Acquisition Module
2102-21FF	Display Memory

Table 11.1
Unibus Addresses

```
;UNIBUS COMMUNICATION TEST
; DISC VERSION  24 MAY 79  B. DONLAN
;
RENT:   EQU     0
;
        ORG     100H
ENTRY3: LXI     H,ENTRY3
        PUSH    H
        LXI     H,MSG5          ;OPENNING MEESSAGE
        CALL    PMSG
        CALL    BBIN            ;GET HEX CHAR
        PUSH    D               ;SAVE ADDRESS
        LXI     H,MSG6          ;REQUEST MODE
        CALL    PMSG
TRYGN:  CALL    CONIN
        CPI     'I'
        JZ      PUTIN           ;JUMP IF INPUT MAODE
        CPI     03H             ;TEST IF CONTROL C
        JZ      RENT            ;RETURN TO MONITOR
        CPI     'O'
        JNZ     QUEST
;
;OUTPUT MODE
;
PUTOUT: LXI     H,MSG11         ;OUTPUT MESSAGE
        CALL    PMSG
        CALL    BBIN            ;GET DIGITS TO OUTPUT
        POP     B               ;RESTORE ADDRESS TO REG B & C
        CALL    DATAO           ;UNIBUSS DRIVER
        JMP     DONE
;
PUTIN:  LXI     H,MSG9          ;INPUT MESAGE
        CALL    PMSG
        POP     B               ;RESTORE ADDRESS TO B & C
        CALL    DATAI           ;UNIBUS INPUT ROUTINE
        CALL    BINB            ;PRINT DATA FROM BUSS
        JMP     DONE
;
;
DONE:   LXI     H,MSG10
        CALL    PMSG            ;PRINT END OF TEST
        JMP     ENTRY3
;
;
QUEST:  LXI     H,MSG7
        CALL    PMSG            ; ??
        JMP     TRYGN
;
;
MSG5:   DB      OAH,OAH,ODH,'UNIBUS COMMUNICATION TEST'
        DB      OAH,ODH,'ENTER UNIBUS ADDRESS    ',0
MSG6:   DB      OAH,ODH,'INPUT (I), OUTPUT (O), EXIT (CONTROL C) ?',0
MSG7:   DB      OAH,ODH,' ?',0
MSG11:  DB      OAH,ODH,'ENTER DATA TO OUTPUT IN 4 HEX DIGITS    ',0
MSG9:   DB      OAH,ODH,' DATA FROM BUS    ',0
MSG10   DB      OAH,ODH,'TRANSFER COMPLETE',0
```

```
; diagnostic input output routines
; for brian donlan  26 feb 79

CSTAT   EQU  3              ;CONSOLE STATUS PORT.
CCOM    EQU  3              ;CONSOLE COMMAND PORT.
CDATA   EQU  2              ;CONSOLE DATA PORT.
CKBR    EQU  00000010B      ;KEYBOARD READY BIT.
CPTR    EQU  00000001B      ;PRINT READY BIT.
CNULL   EQU  1              ;CONSOLE NULL COUNT.

; CHECK CONSOLE INPUT STATUS.
;

CONST:  IN   CSTAT          ;READ CONSOLE STATUS.
        ANI  CKBR           ;LOOK AT KB READY BIT.
        MVI  A,0            ;SET A=0 FOR RETURN.
        RZ                  ;NOT READY WHEN ZERO.
        CMA                 ;IF READY A=FF.
        RET                 ;RETURN FROM CONST.

;
; READ A CHARACTER FROM CONSOLE.
;
CONIN:   IN   CSTAT                 ;READ CONSOLE STATUS.
         ANI  CKBR          ;IF NOT READY,

        JZ   CONIN          ;READY WHEN HIGH.
         IN   CDATA         ;READ A CHARACTER.
        OUT     CDATA
         ANI  7FH           ;MAKE MOST SIG. BIT = 0.
         RET
;
; WRITE A CHARACTER TO THE CONSOLE DEVICE.
;
CONOT:  MVI  A,0DH          ;IF IT'S A CR,
        CMP  C              ;THEN HOP OUT
        JZ   CONUL          ;TO NULL ROUTINE.
CONOT1: IN   CSTAT          ;READ CONSOLE STATUS.
        ANI  CPTR           ;IF NOT READY,
        JZ   CONOT1         ;READY WHEN HIGH.
        MOV  A,C            ;GET CHARACTER.
        OUT  CDATA          ;PRINT IT.
        RET                 ;RETURN.
CONUL:  PUSH B              ;SAVE B&C.
        MVI  B,CNULL        ;GET NULL COUNT.
CONUL1: CALL CONOT1         ;PRINT CR.
        MVI  C,0            ;GET NULL CHAR.
        DCR  B              ;DECREMENT COUNTER.
        JNZ  CONUL1         ;DO NEXT NULL.
        POP  B              ;RESTORE B&C.
        MOV  A,C            ;RESTORE A.
        RET                 ;RETURN.
;

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;       PRINT MESSAGE UNTIL ZERO
;       MESSAGE ADDRESS REG H & L
;
;;;;;;;;;;;;;;;;;;;;;
PMSG:   MOV     A,M     ;GET CHAR
        ORA     A       ;IS IT A  ZERO
        RZ
        MOV     C,A     ;OTHERWISE PRINT
        CALL    CONOT
        INX     H       ;INC ADDRESSS
        JMP     PMSG
```

```
;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;       PRINT 8 BIT WORD IN BINARY FORMAT
;           INPUT:  DATA IN REG A
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;
BITS:     MOV       B,A        ; DATA
          MVI       A,80H      ; MASK
OVER:     MVI       C,30H
          MOV       E,A        ; STORE MASK
          ANA       B          ; AND WITH MASK
          JZ        PRNT       ; JUMP IF ZERO
          MVI       C,31H
PRNT:     CALL      CONOT
          ANA       B          ; ZERO CARRY
          MOV       A,E        ; LOAD MASK
          RAR
          JNC       OVER
          RET
;;
;
BLNK:     MVI       C,20H                  ;PRINT BLANKS, # IN REG. D
LP1:      CALL      CONOT1
          DCR       D
          JNZ       LP1
          RET
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;
;       OUTPUTS 2 HEX DIGITS IN ASCCII
;           FROM REG D
;
;;;;;;;;;;;
;
;
BINHA:    MOV       A,D
          RAR
          RAR
          RAR
          RAR
          CALL      BIN1
          MOV       C,A
          CALL      CONOT
          MOV       A,D
          CALL      BIN1
          MOV       C,A
          CALL      CONOT
          RET
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;    OUTPUTS FOUR HEX DIGITS IN ASCII
;    ENTER WITH DATA IN REG PAIR E AND D
;;;;;;;;;;;;;;;;;;;;;;
;;
;
BINB:     CALL      BINHA
          MOV       D,E
          CALL      BINHA
          RET
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;    CONVERTS HEX TO ASCII
;    INPUT: 4 BITS HEX  REG A
;    OUTPUT:  8 BIT ASSCII REG A
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;
;
BIN1:     ANI       0FH
          ADI       30H
          CPI       3AH
          RC
          ADI       07H
          RET
```

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;     INPUT 4 DIGITS FROM CONSOLE
;     RETURN;  4 HEX DIGITS IN REG E-D
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;
;
BBIN:   CALL    CONIN
        CALL    AHS1
        RAL
        RAL
        RAL
        RAL
        ANI     0F0H
        MOV     D,A
        CALL    CONIN
        CALL    AHS1
        ANI     0FH
        ORA     D
        MOV     D,A
        CALL    CONIN
        CALL    AHS1
        RAL
        RAL
        RAL
        RAL
        ANI     0F0H
        MOV     E,A
        CALL    CONIN
        CALL    AHS1
        ANI     0FH
        ORA     E
        MOV     E,A
        RET
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;     CONVERT ASCII TO HEX
;     INPUT;  8 BIT ASCII  REG A
;     OUTPUT:  4 BIT HEX REG A
;
;;;;;;;;;;;;;;;;;;;;;;;;;
;;
;
AHS1:   NOP
        SUI     30H
        CPI     0AH
        RC
        SUI     07H
        RET
;
;;;;;;;;;;
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;      INITIATE SIO PORTS
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;
;
INITA:  MVI     A,0AAH          ;GET DUMMY MODE WORD
         OUT    CSTAT           ;OUTPUT IT
        MVI     A,40H           ;GET RESET BIT
        OUT     CSTAT           ;RESET SIO BOARD
        MVI     A,0CEH          ;GET REAL MODE WORD
        OUT     CSTAT           ;SET THE MODE FOR REAL
        MVI     A,37H           ;GET THE COMMAND
        OUT     CSTAT           ;OUTPUT IT
        RET
;
;
;
CRLF:   MVI     C,13            ;CR
        CALL    CONOT
LF:     MVI     C,10            ;LF
        CALL    CONOT1
        MVI     C,7FH
        CALL    CONOT1
        CALL    CONOT
        RET
```

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;
;
DATAI:  CALL    GETBUS
        CALL    DATI
        SUB     A
        OUT     15H
        RET
;
;
DATAO:  CALL    GETBUS
        CALL    DATO
        SUB     A
        OUT     15H
        RET
;
;
;
;ROUTINE TO INPUT A 15 BIT WORD FROM UNIBUS
;REG B = A<16:09>, REG C = A<08:01>
;DATA WILL BE CONTAINED IN REG D = D<15:08>, REG C = D<07:00>
;
;
DATI:   MVI     A,0FFH          ;SET LOOP COUNT
        STA     BIZCNT
BIZLP1: IN      14H             ;CHECK FOR SYS = 0
        ANI     04H             ;FROM LAST TRANSACTION
        JNZ     BBUSY1
;
        MOV     A,B             ;OUTPUT HIGH ADDRESS
        OUT     10H
        MOV     A,C             ;OUTPUT LOW ADDRESS
        OUT     11H
        SUB     A               ;OUTPUT C1=0
        OUT     14H
        ORI     01H             ;OUTPUT MSYN=1
        OUT     14H
;
SYNLP1: MVI     A,0FFH          ;LOOP COUNT
        STA     SYNCNT
DILOOP: IN      14H             ;CHECKS IF SSYN=1
        OUT     0FFH
        ANI     04H
        JZ      NOSYN1
;
        IN      12H             ;INPUT HIGH DATA
        MOV     D,A
        IN      13H             ;INPUT LOW DATA
        MOV     E,A
;
        SUB     A
        OUT     14H             ;CLEARS MSYN
        OUT     10H             ;AND EVERYTHING
        OUT     11H             ;PUT OUT TO BUS
        OUT     12H
        OUT     13H
        IN      0FFH
        ANI     080H
        JNZ     DATI            ;LOOP IF SENSE SWITCH UP
        RET
```

```
;
;ROUTINE TO OUTPUT A 16 BIT WORD ON THE UNIBUS
;REG B = A(15-08), REG C = A(07-00)
;REG D = D(15-08), REG E = D(07-00)
;
DATO:   MVI     A,0FFH
        STA     BIZCNT
BIZLP2: IN      14H
        ANI     04H
        JNZ     BBUSY2
;
        MOV     A,B                     ;OUTPUT HIGH ADDRESS
        OUT     10H
        MOV     A,C                     ;OUTPUT LOW ADDRESS
        OUT     11H
        MOV     A,D                     ;OUTPUT HIGH DATA
        OUT     12H
        MOV     A,E                     ;OUTPUT LOW DATA
        OUT     13H
        MVI     A,02H                   ;OUTPUT C1=1
        OUT     14H
;
        MVI     A,03H                   ;OUTPUT MSYN=1
        OUT     14H
;
SYNLP2: MVI     A,0FFH
        STA     SYNCNT
DOLOOP: IN      14H                     ;CHECKS FOR SSYN
        OUT     0FFH
        ANI     04H                     ;TO GET ASSERTED
        JZ      NOSYN2
;
        SUB     A
        OUT     14H                     ;CLEARS MSYN AND C1
        OUT     10H
        OUT     11H                     ;CLEARS EVERYTHING
        OUT     12H                     ;OUTPUT TO THE BUS
        OUT     13H
        IN      0FFH                    ;READ SENSE SWITCH
        ANI     080H
        JNZ     DATO                    ;LOOP IF UP
        RET
;
;
;
;
GETBUS: MVI     A,0FFH
        STA     GETCNT
        MVI     A,01H
        OUT     15H
LOOP:   IN      15H
        ANI     01H
        JZ      NOGET
        RET
;  ON-LINE UNIBUS DIAGNOSTICS
;    BY BRIAN DONLAN  24 APR 79
;
BBUSY1: LDA BIZCNT                      ;LOOP COUNT
        DCR     A
        STA     BIZCNT                  ;NEW COUNT
        JNZ     BIZLP1                  ;JUMP IF STILL COUNT
        LXI     H,ERMSG2
        CALL    PMSG    ;DISPLAY ERROR MESSAGE
        JMP     ENTRY3
;
;
BBUSY2: LDA     BIZCNT
        DCR     A
        STA     BIZCNT
        JNZ     BIZLP2
        LXI     H,ERMSG2
        CALL    PMSG
        JMP     ENTRY3
```

```
;
NOSYN1: LDA     SYNCNT
        DCR     A
        STA     SYNCNT
        JNZ     DILOOP
        IN      0FFH
        ANI     040H
        JNZ     DILOOP
SYSERR: MOV     D,B                     ;
        MOV     E,C                     ;MOV ADDRESS FOR OUTPUT
        LXI     H,ERMSG3
        CALL    PMSG
        CALL    BINB                    ;OUTPUT ADDRESS
        LXI     H,ERMSG4
        CALL    PMSG
        XRA     A                       ;ZERO A
        OUT     11H
        OUT     12H
        OUT     13H
        OUT     14H
        OUT     10H
        JMP     ENTRY3
;
NOSYN2: LDA     SYNCNT
        DCR     A
        STA     SYNCNT
        JNZ     DOLOOP
        IN      0FFH
        ANI     040H
        JNZ     DOLOOP
        JMP     SYSERR
;
;
;
;
NOGET:  LDA     GETCNT
        DCR     A
        STA     GETCNT
        JNZ     LOOP
        LXI     H,ERMSG1
        CALL    PMSG
        JMP     ENTRY3
;
;
;
;
;
;
GETCNT: DS      1
SYNCNT: DS      1
BIZCNT: DS      1
ERMSG1: DB      0DH,0DH,0AH,'  IMSAI CAN NOT GET BUSS    ',0
ERMSG2: DB      0DH,0DH,0AH,'  ERROR BUSS BUSY          ',0
ERMSG3: DB      0DH,0AH,'  DEVICE NO. ',0
ERMSG4: DB      ' NO RESPONSE ',0
;
;
        END
```

Section 12

Unibus Snapshot

12.1 General Description

The Unibus Snapshot program is not a test, but a routine which presents on the console device the status of the Unibus data, address and some status lines at the time the routine was executed.

12.2 Program Details

The Unibus Snapshot routine can either be run as a stand-alone program, or by using an alternate entry point as a subroutine which can be called by any user program. The Snapshot routine presents in Hex, the status of the Unibus address and data lines. The routine also presents in binary, the Unibus slave sync and bus grant lines. The routine also has the capability to present the master sync line, but this requires some minor hardware changes which are not available at this time.

12.3 Operation

Snapshot requires no operator responses or input to run and upon completion of the listing, execution is returned to the monitor or program from which it came.

Snapshot is stored in programmable memory on the diagnostic memory board. The program can be run under the diagnostic operating system test controller, or it can be run from the IMSAI 8080 front panel at a starting addr-

AD-A092 441 AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH F/G 14/2
DIAGNOSTIC SOFTWARE DEVELOPMENT FOR A REAL-TIME SPECTRAL ANALYS--ETC(U)
APR 80 B J DONLAN
UNCLASSIFIED AFIT-CI-80-9T NL

2 OF 2
AD
A092441

END
DATE
FILMED
1-81
DTIC

of 'D500' Hex.

Snapshot is also stored on floppy disk and it can be invoked by the CPM operating system under file name 'SNAPST.COM'.

Snapshot is also available as a subroutine which can be called from user program. The PROM version subroutine entry point is at 'D508' Hex.

```
; UNIBUS SNAP SHOT ROUTINE
        ORG     100H
ENTRY4: LXI     H,FINIS
        PUSH    H
        DI
        CALL    INITA           ;RESET I/O
;
;SUBROUTINE ENTRY POINT
ENTRY5: LXI     H,MSG12
        CALL    PMSG
        IN      10H             ;HIGH ADDRESS
        MOV     D,A             ;SAVE IN D
        IN      11H             ;LOW ADDRESS
        MOV     E,A
        CALL    BINB            ;PRINT UNIBUS ADDRESS
;
        MVI     D,8H            ;SPACE OVER
        CALL    BLNK
        IN      12H             ;HIGH DATA
        MOV     D,A
        IN      13H             ;LOW DATA BITS
        MOV     E,A
        CALL    BINB            ;PRINT UNIBUS DATA BITS
        MVI     D,8H            ;SPACE OVER
        CALL    BLNK
;
        IN      14H             ;STATUS PORT
        ANI     04H             ;FIND SLAVE SYN
        JZ      NOSIS
        MVI     C,'1'
        JMP     OUTSIS
NOSIS:  MVI     C,'0'
OUTSIS: CALL    CONOT           ;PRINT SLAVE SYN
        MVI     D,09H           ;SPACE OVER
        CALL    BLNK
;
        IN      15H             ;STATUS PORT
        ANI     01H             ;BUS GRANT
        JZ      NOBUS
        MVI     C,'1'
        JMP     OUTBUS
NOBUS:  MVI     C,'0'
OUTBUS: CALL    CONOT           ;PRINT BUS GRANT
        MVI     D,0BH           ;SPACE OVER
        CALL    BLNK
;
        IN      14H
        ANI     00H
        JZ      NOMSYN
        MVI     C,'1'
        JMP     OUTMSN
NOMSYN: MVI     C,'0'
OUTMSN: CALL    CONOT           ;PRINT MSYN
        RET
FINIS:  JMP     FINIS
;
MSG12:  DB      OAH,OAH,ODH,'UNIBUS SNAP-SHOT  '
        DB      OAH,ODH,'ADDRESS        DATA      SSYN      GRANT      MSYN'
        DB      OAH,ODH,'  ',0
;
; for brian donlan  26 feb 79

CSTAT   EQU  3          ;CONSOLE STATUS PORT.
CCOM    EQU  3          ;CONSOLE COMMAND PORT.
CDATA   EQU  2          ;CONSOLE DATA PORT.
CKBR    EQU  00000010B  ;KEYBOARD READY BIT.
CPTR    EQU  00000001B  ;PRINT READY BIT.
CNULL   EQU  1          ;CONSOLE NULL COUNT.
```

```
; CHECK CONSOLE INPUT STATUS.
;
CONST:  IN   CSTAT       ;READ CONSOLE STATUS.
        ANI  CKBR        ;LOOK AT KB READY BIT.
        MVI  A,0         ;SET A=0 FOR RETURN.
        RZ               ;NOT READY WHEN ZERO.
        CMA              ;IF READY A=FF.
        RET              ;RETURN FROM CONST.

;
; READ A CHARACTER FROM CONSOLE.
;
CONIN:   IN   CSTAT              ;READ CONSOLE STATUS.
         ANI  CKBR       ;IF NOT READY,

        JZ   CONIN       ;READY WHEN HIGH.
         IN   CDATA      ;READ A CHARACTER.
        OUT     CDATA
         ANI  7FH        ;MAKE MOST SIG. BIT = 0.
         RET
;
; WRITE A CHARACTER TO THE CONSOLE DEVICE.
;
CONOT:  MVI  A,0DH       ;IF IT'S A CR,
        CMP  C           ;THEN HOP OUT
        JZ   CONUL       ;TO NULL ROUTINE.
CONOT1: IN   CSTAT       ;READ CONSOLE STATUS.
        ANI  CPTR        ;IF NOT READY,
        JZ   CONOT1      ;READY WHEN HIGH.
        MOV  A,C         ;GET CHARACTER.
        OUT  CDATA       ;PRINT IT.
        RET              ;RETURN.
CONUL:  PUSH B           ;SAVE B&C.
        MVI  B,CNULL     ;GET NULL COUNT.
CONUL1: CALL CONOT1      ;PRINT CR.
        MVI  C,0         ;GET NULL CHAR.
        DCR  B           ;DECREMENT COUNTER.
        JNZ  CONUL1      ;DO NEXT NULL.
        POP  B           ;RESTORE B&C.
        MOV  A,C         ;RESTORE A.
        RET              ;RETURN.
;

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;       PRINT MESSAGE UNTIL ZERO
;       MESSAGE ADDRESS REG H & L
;
;;;;;;;;;;;;;;;;;;;;;
PMSG:   MOV     A,M      ;GET CHAR
        ORA     A        ;IS IT A  ZERO
        RZ
        MOV     C,A      ;OTHERWISE PRINT
        CALL    CONOT
        INX     H        ;INC ADDRESSS
        JMP     PMSG
;
;
;
;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;    PRINT 8 BIT WORD IN BINARY FORMAT
;       INPUT:  DATA IN REG A
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;
BITS:   MOV     B,A      ; DATA
        MVI     A,80H    ; MASK
OVER:   MVI     C,30H
        MOV     E,A      ; STORE MASK
        ANA     B        ; AND WITH MASK
        JZ      PRNT     ; JUMP IF ZERO
        MVI     C,31H
```

```
PRNT:   CALL    CONOT
        ANA     B       ; ZERO CARRY
        MOV     A,E     ; LOAD MASK
        RAR
        JNC     OVER
        RET
;;
;
BLNK:   MVI     C,20H           ;PRINT BLANKS, # IN REG. D
LP31:   CALL    CONOT1
        DCR     D
        JNZ     LP31
        RET
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;
;     OUTPUTS 2 HEX DIGITS IN ASCCII
;            FROM REG D
;
;;;;;;;;;;;;
;
;
BINHA:  MOV     A,D
        RAR
        RAR
        RAR
        RAR
        CALL    BIN1
        MOV     C,A
        CALL    CONOT
        MOV     A,D
        CALL    BIN1
        MOV     C,A
        CALL    CONOT
        RET
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;    OUTPUTS FOUR HEX DIGITS IN ASCII
;    ENTER WITH DATA IN REG PAIR E AND D
;;;;;;;;;;;;;;;;;;
;;
;
BINB:   CALL    BINHA
        MOV     D,E
        CALL    BINHA
        RET
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;    CONVERTS HEX TO ASCII
;    INPUT: 4 BITS HEX  REG A
;    OUTPUT:  8 BIT ASSCII REG A
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;
;
BIN1:   ANI     0FH
        ADI     30H
        CPI     3AH
        RC
        ADI     07H
        RET
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;    INPUTS 4 DIGITS FROM CONSOLE
;    RETURN;  4 HEX DIGITS IN REG E-D
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;
;
BBIN:   CALL    CONIN
        CALL    AHS1
        RAL
        RAL
        RAL
        RAL
        ANI     OFOH
        MOV     D,A
        CALL    CONIN
        CALL    AHS1
        ANI     OFH
        ORA     D
        MOV     D,A
        CALL    CONIN
        CALL    AHS1
        RAL
        RAL
        RAL
        RAL
        ANI     OFOH
        MOV     E,A
        CALL    CONIN
        CALL    AHS1
        ANI     OFH
        ORA     E
        MOV     E,A
        RET
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;    CONVERT ASCII TO HEX
;    INPUT;  8 BIT ASCII  REG A
;    OUTPUT:  4 BIT HEX REG A
;
;;;;;;;;;;;;;;;;;;;;;;;;;;
;;
;
AHS1:   NOP
        SUI     30H
        CPI     OAH
        RC
        SUI     07H
        RET
;
;;;;;;;;;;
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;     INITIATE SIO PORTS
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;
;
INITA:  MVI     A,OAAH                  ;GET DUMMY MODE WORD
         OUT    CSTAT                   ;OUTPUT IT
        MVI     A,40H                   ;GET RESET BIT
        OUT     CSTAT                   ;RESET SIO BOARD
        MVI     A,OCEH                  ;GET REAL MODE WORD
        OUT     CSTAT                   ;SET THE MODE FOR REAL
        MVI     A,37H                   ;GET THE COMMAND
        OUT     CSTAT                   ;OUTPUT IT
        RET
;
;
;
CRLF:   MVI     C,13                    ;CR
        CALL    CONOT
LF:     MVI     C,10                    ;LF
        CALL    CONOT1
        MVI     C,7FH
        CALL    CONOT1
        CALL    CONOT
        RET
```

Secton 13

Diagnostic Memory Board

13.1 General Description

In order to have all diagnostic programs available for execution even in the event of a disk failure, all diagnostic programs were placed in programmable read-only memories. A small operating system was also included in the prom memory. This enabled all programs to be resident and not require any disk loading before execution. A 16K byte read-only memory board was added to the IMSAI 8080 computer system to hold these programs.

13.2 Detailed Description

The memory board decided upon was in a kit manufactured by SSM Microcomuter products. This board has a capacity of 16K bytes of memory stored in Intel 2708 memory chips. The memory board can be assigned to any 16K memory block area but for this application it was placed at the very top of the memory address range. The board was set to occupy from 'C000' to 'FFFF' hex. Presently. only 10K of the possible 16K is used for diagnostics, leaving the remainder for future expansion.

The memory board is S-100 Bus compatable and receives all its power from the Bus.

'C000'	Memory Test	
'C290'	Mini Memory test	0 to 100 Hex version
'C800'	Formatted Disc test	
'CD80'	Disc Full track write routine	
'CE40'	Disc Full track write routine	
'D000'	Unibus Port test	
'D100'	Unibus Communication test	
'D500'	Unibus Snap Shot routine	
'D600'	Mini-Memory test	8K version
'D700'	Mini-Memory test	24K version
'F000'	Operating system	
'F800'	Help program	
'FD00'	test controller	
'E000'	Color graphic test (future options)	

table 13.1 Memory map

Section 14

Conclusion

This report has described the design and development of a diagnostic system for a real-time spectral analysis system.

The developed software has been verified and tested. In fact, many of the tests were used by the author and other members of the project team to keep the system operational, enabling further development to proceed. The diagnostic system also proved very helpful in testing new hardware designs and components. For example, hardware modifications were performed on the color graphics display, giving it added capabilities. During the development of a display test, the newly written programs pointed out unknown degradations of the original capabilities.

Since the research personnel working on and with the spectral analysis system is always changing, every effort was made to make the tests easy to run and error notifications self-explanatory. This is unfortunate since the tests contain a wealth of information about the system state and a more experienced user can interpret this information.

The structure of the diagnostic system is such that many of the functions are performed using subroutines. These subroutines can be used by any program and could be of great use in the future for I/O functions on on-line continuous verification.

Unfortunately, as of this writing, no further funding or development effort is programmed of the system. With the exception of the memory diagnostics. all the tests are very specific and will be of little value for use in a general purpose microcomputer system.

The diagnostic system was designed with expansion in mind. The programmable memory board contains room for many more tests and the test directory can easily be expanded.

Section 15

Literature

1. Lee, Bock w., Two Additions to the Z-80 RAM Test, Dr. Dobb's Journal, Jan 1979.

2. Knaizuk, John, Hartmann, C.P.R., An Algorithm for Testing Random Access Memories, IEEE Trans. on Computers, April 1977.

3. Intel Corp., Intel 8080 Microcomputer User's Manual, Intel Corp., Santa Clara, CA., 1975.

4. IMSAI Assco., IMSAI 8080 Microcomputer User's Manual Set, IMSAI Manf. Corp., San leandro, CA., 1976.

5. Ng, Howard, Disk System for a Microcomputer Controlled Spectra Analysis System, Master's Project, RPI, 1978

6. Tarbell Electronics, The Tarbell Floppy Disk Interface, Tarbell Electronics, Carson, CA., 1977

7. Dykstra, K.U., Input Module and Front Panel Design for a Real Time Spectral Analysis System, Master's Project, RPI, 1978.

8. Digital Research, CP/M Technical Manual Set, Digital Research, Pacific Grove, CA., 1976, 1977.

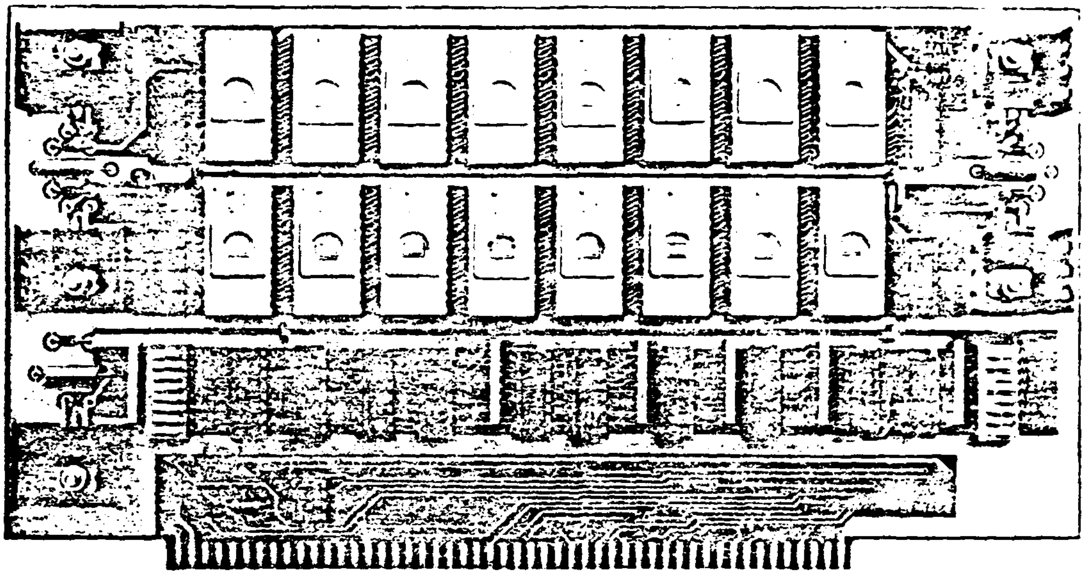
9. Sparrell, D.K., Software Development for a Real-Time Spectral Analysis System for Nonstationary Signals, Master's Project, RPI, 1978.

10. Scordo, Dominick, A Real-Time Signal Processing and Display System for Non-Stationary Signals, Master's Project, RPI, 1978.

APPENDIX A

MEMORY BOARD DIAGRAMS

MB8A 1K TO 16K EPROM BOARD



FEATURES:

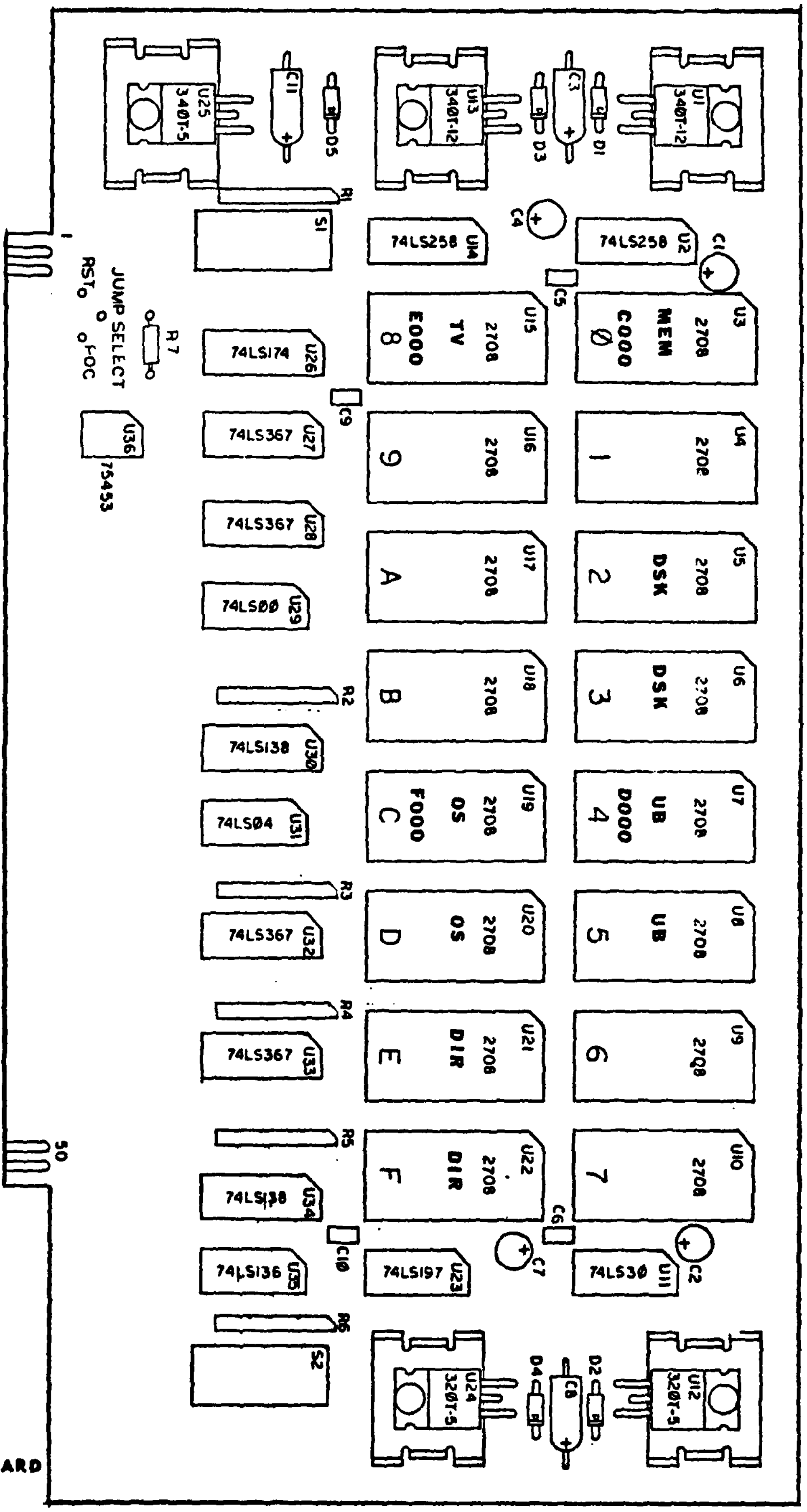
SYSTEM COMPATIBILITY
- S-100 bus computer systems.

MEMORY
- Up to 16K bytes of 2708 EPROMs (not included)
- Any unused EPROM socket will automatically disable the board for that 1K increment. For example, with 8 EPROMs it acts as an 8K board, taking up only 8K of memory address space.

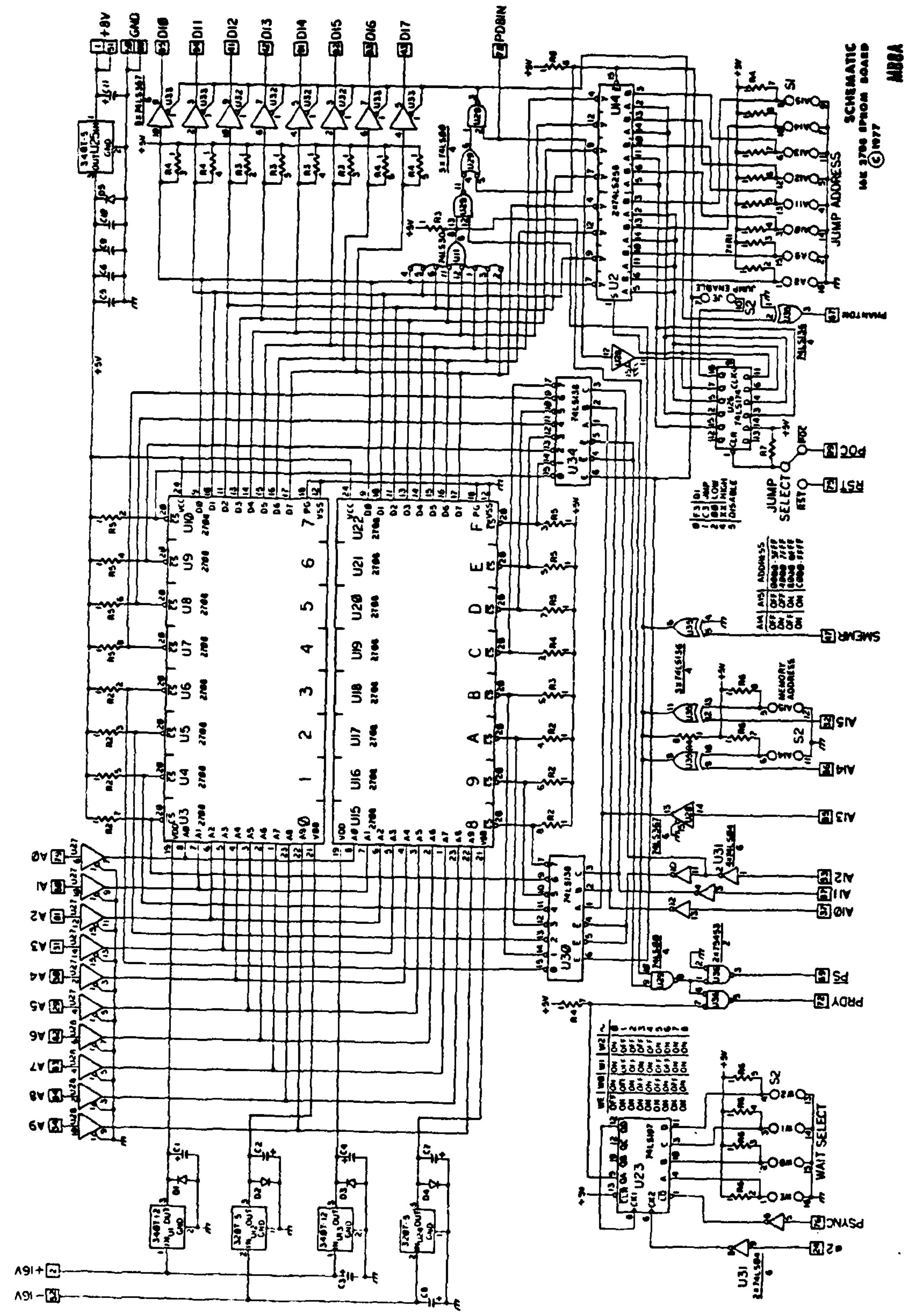
ADDRESSING
- DIP switch selection of memory address assignment in 16K byte increments.
- Magic Mapping TM allows any byte within ROM to be mixed with any similarly addressed RAM board equipped with Phantom Disable.

VECTOR JUMP
- Power-on/reset vector jump to any 256 byte increment; DIP switch addressable.
- Vector jump can be disabled.
- Vector jump requires other memory boards to be equipped with Phantom Disable.

OTHER FEATURES
- DIP switch selection of 0 to 8 wait state clock cycles, so fast or slow EPROMs can be used.
- All lines buffered, Reverse voltage protection.
- High grade glass epoxy PC board with gold plated edge connector contacts.
- Low profile sockets provided for all ICs.
- Power requirements (less EPROMs) -- +8V @ 160mA, +16V @ 10mA, -16V @ 10mA typical.



ASSEMBLY
16K 2708 EPROM BOARD
© 1977
MB8A


SCHEMATIC
16K 2708 EPROM BOARD
©1977
MBBA
JUMP ADDRESS
JUMP SELECT
JUMP ENABLE
MEMORY ADDRESS
WAIT SELECT
+8V
GND
DI0
DI1
DI2
DI3
DI4
DI5
DI6
DI7
PDBIN
PHANTOM
POC
RST
SMEMR
A15
A14
A13
A12
A11
A10
PS
PRDY
PSYNC
φ2
A0
A1
A2
A3
A4
A5
A6
A7
A8
A9
+16V
-16V
U2
U3
U4
U5
U6
U7
U8
U9
U10
U14
U15
U16
U17
U18
U19
U20
U21
U22
U23
U26
U30
U31
U34
S1
S2
2708
74LS138
74LS175
74LS197
74LS257
1
2
3
4
5
6
7
8
9
A
B
C
D
E
F
0

APPENDIX B

COMMONLY USED SUBROUTINES

Commonly Used Subroutine

The following is a listing of some subroutines which are commonly used in the diagnostic programs:

CONST 'C177'

Checks console status. Returns with zero in reg A if not ready. Returns with 'FF' in reg A if ready.

CONIN 'C180'

Reads a character from the console. The input character is returned in reg A. The input character is echoed on the console.

CONOT 'C18E'

Writes a character on the console device. The character is output from reg C.

PMSG 'C1AE'

Prints a character string on the console device. The address of the beginning of the string must be placed in reg H and L. The string is printed until a null (00) is encountered.

BITS 'C1E5'

Prints an 8 bit byte in binary format in the console device. The data word is taken from reg A.

BLNK 'C1FB'

Prints the number of blanks found in reg D.

BINB 'C21A'

Outputs four hex digits in ASCII on the console. Enter with the data in reg E and D.

BINHA 'C205'

Outputs two hex digits in ASCII on the console. Enter with the data in reg D.

BIN1 'C222'

Converts hex to ASCII. Input with 4 bits in reg A Outputs with 8 bit ASCII character in reg A.

BBIN 'C22C'

Inputs 4 hex digits from the console. Converts the ASCII characters to hex. Returns with the 4 hex digits in reg E and D.

AHS1 'C25B

Converts ASCII to Hex. Inputs with a 8bit ASCII character in reg A. Returns with a 4 bit hex digit in reg A.

INITA 'C264'

Initiates the SIO port. No inputs or outputs.

CRLF 'C275'

Sends one Carriage return and one line feed to the console.

LF 'C27A'

Sends one line feed to the console.

DATI 'D312'

Inputs a 16 bit word from the Unibus. Reg B and C needs the Unibus address and the data will be returned in reg D and E.

DATO 'D351'

Outputs a 16 bit word to the Unibus. Reg B and C needs the Unibus address and reg D and E needs the data.

GETBUS 'D391'

This routine gets the IMSAI master-ship of the unibus.

APPENDIX C

PROM PROGRAM LISTINGS

```
                        ;       MEMORY TEST
                        ;    PROM VERSION 24 MAY 79  B. DONLAN
                        ;
C000                            ORG     0C000H
                        ;
                        ;
0000 =                  WO      EQU     00      ;TEST BYTE
                        ;
C000 218000             ENTRY1: LXI     H,080H
C003 F9                         SPHL
C004 2104C0             ENTRY:  LXI     H,ENTRY
C007 E5                         PUSH    H
C008 3E00                       MVI     A,00            ;ZERO ACC
C00A 320A00                     STA     CODE
C00D CD64C2                     CALL    INITA           ;RESET I/O PORT
C010 CD75C2                     CALL    CRLF
C013 2120C1                     LXI     H,MSG1
C016 CDAEC1                     CALL    PMSG
C019 212EC1                     LXI     H,MSG2
C01C CDAEC1                     CALL    PMSG
C01F CD2CC2                     CALL    BBIN
C022 EB                         XCHG
C023 220400                     SHLD    START
C026 2149C1                     LXI     H,MSG3
C029 CDAEC1                     CALL    PMSG
C02C CD2CC2                     CALL    BBIN
C02F EB                         XCHG
C030 220600                     SHLD    ENADR
C033 21B9C1                     LXI     H,MSG8
C036 CDAEC1                     CALL    PMSG
C039 DB02                       IN      CDATA           ;RESET IO FLAG
                        ;
                        ;
C03B 0E00               BEGIN:  MVI     C,WO    ;LOAD TEST BYTE
C03D 3E02               MTEST:  MVI     A,02    ;LOAD TEST BYTE
C03F 320800                     STA      PART
C042 CD8EC0             MTLOP:  CALL    STUFF   ;STUFF MAJOR ALL OVER
C045 3E02                       MVI     A,02    ;SET TWO AS MINOR
C047 CD98C0                     CALL    STUFM   ;STUFF MINOR
C04A 3E02                       MVI     A,02    ; SET 2 AGAIN
C04C CDB0C0                     CALL    CHECK   ;NOW CHECK ALL LOC
C04F 3A0800             PTCHK:  LDA     PART
C052 3D                         DCR     A
C053 320800                     STA     PART    ;STORE NEW PART
C056 FE00                       CPI     00      ;FINISH THIS PASS ?
C058 CA6AC0                     JZ      RECYCLE ;YES
C05B 3E01               CONT:   MVI     A,01    ;NO CONTINUE
C05D CD98C0                     CALL    STUFM   ;STUFF MINOR SERT
C060 79                         MOV     A,C     ;LOAD MAJOR BYTE
C061 2F                         CMA             ;COMPLIMENT MAJOR BYTE
C062 4F                         MOV     C,A     ;SAVE NEW BYTE
C063 AF                         XRA     A       ;ZERO OTHER TEST BYTE
C064 CDB0C0                     CALL    CHECK
C067 C342C0                     JMP     MTLOP
                        ;
                        ;
                        RECYCLE:
C06A 79                         MOV     A,C
C06B 320800                     STA     PART    ;SAVE INVERT TB TEMP
C06E 2162C1                     LXI     H,MSG4  ;END OF PAS MESSAGE
C071 CDAEC1                     CALL    PMSG
C074 3A0A00                     LDA     CODE    ;CHAR CODE
C077 FE03                       CPI     03H     ;CONTROL C
C079 CA21F0                     JZ      RENT    ;RETURN TO MONITRO
C07C B7                         ORA     A       ;SET FLAGS
C07D C204C0                     JNZ     ENTRY   ;START OVER
C080 A7                         ANA     A       ;CLEAR CARRY
C081 3A0800                     LDA     PART    ; RECOVER TEST BYTE
C084 B7                         ORA     A
C085 CA3BC0                     JZ      BEGIN
C088 17                         RAL
C089 2F                         CMA
C08A 4F                 TB:     MOV     C,A     ;NEW TEST BYTE
C08B C33DC0                     JMP     MTEST   ; ANOTHER PAS
```

```
                      ;
F021 =                RENT:    EQU     OF021H   ;MONITOR ENTRY
0004 =                START:   EQU     04H      ;LOC OF START ADDR
0006 =                ENADR:   EQU     06H      ;LOC OF END ADDR
0008 =                PART:    EQU     08H      ;LOC FOR PART
000A =                CODE:    EQU     0AH      ;LOC FOR CODE
                      ;
                      ;
C08E CDDEC0           STUFF:   CALL    STASTO   ;LOAD START AND END ADDR
C091 71               DOIT:    MOV     M,C      ;STUFF MAJOR ALL OVER
C092 CD06C1                    CALL    HILOX    ;SEE IF ALL MEM DONE
C095 C391C0                    JMP     DOIT     ;NO KEPP ON STUFFING
                      ;
                      ;
                      ;
C098 CDDEC0           STUFM:   CALL    STASTO   ;LOAD ADDR AGAIN
C09B 47                        MOV     B,A      ;MINOR COUNTER
C09C FE00                      CPI     00       ;MINOR WORD STUFF
C09E C2A6C0                    JNZ     HIL      ;NO
C0A1 79               MINOR:   MOV     A,C      ;MAJOR TEST BYTE
C0A2 2F                        CMA              ;MINOR IS COMPLIMENT OF MAJOR
COA3 77                        MOV     M,A      ;STUFF MINOR BYTE IN MEM
COA4 0603                      MVI     B,03     ;START MINOR COUNT AT 3
COA6 CD06C1           HIL:     CALL    HILOX    ;INC & CHK IF DONE
COA9 05                        DCR     B        ;DEC MINOR COUNTER
COAA C2A6C0                    JNZ     HIL      ;OK TO STUFF NO
COAD C3A1C0                    JMP     MINOR    ;YES
                      ;
                      ;
COB0 CDDEC0           CHECK:   CALL    STASTO            ;LOAD START AND END
COB3 47                        MOV     B,A      ;LOAD MINOR COUNT
COB4 FE00                      CPI     00       ;COUNT ZERO
COB6 C2C1C0                    JNZ     MAJR     ;NO GO TO MAJOR
COB9 79               MINR:    MOV     A,C      ;LOAD TEST BYTE MAJOR
COBA 2F                        CMA              ; MINOR IS COMPLIMENYT
COBB BE                        CMP     M        ;READ AND COMPARE MEM LOC
COBC 0603                      MVI     R,03     ;MINOR COUNT AT 3
COBE C3C3C0                    JMP     CKEND    ;CHECK FOR ERROR OR ABORT
COC1 79               MAJR:    MOV     A,C      ;LOAD MAJOR TEST BYTE
COC2 BE                        CMP     M        ;READ AND COMPARE MEM WITH MAJOR
COC3 C5               CKEND:   PUSH    B        ;SAVE COUNT AND MAJOR
COC4 C4E6C0                    CNZ     ERR      ;GO TO ERR TO PRNT IF ERROR
COC7 C1                        POP     B        ;RESTORE REGS
COC8 DB03                      IN      CSTAT    ;CHECK KEYBOARD
COCA E602                      ANI     02H
COCC CAD4C0                    JZ      FIN
COCF DB02                      IN      CDATA    ;READ KEYS
COD1 320A00                    STA     CODE
COD4 CD06C1           FIN:     CALL    HILOX
COD7 05                        DCR     B        ;DEC MINOR COUNT
COD8 C2C1C0                    JNZ     MAJR
CODB C3B9C0                    JMP     MINR     ;COUNT ZERO DO MINOR
                      ;
                      ;
                      ;
                      ;
CODE 2A0600           STASTO:  LHLD    ENADR    ;LOAD END ADDR
COE1 EB                        XCHG             ;MOVE END TO C&D
COE2 2A0400                    LHLD    START    ;LOAD START
COE5 C9                        RET
```

```
                      ;
 C0E6 D5              ERR:    PUSH    D         ;SAVE END ADDR
 C0E7 F5                      PUSH    PSW
 C0E8 CD75C2                  CALL    CRLF
  0EB 54                      MOV     D,H
 C0EC 5D                      MOV     E,L
 C0ED CD1AC2                  CALL    BINB      ;OUTPUT BAD ADDR
 C0F0 1608                    MVI     D,08      ;SPACE COUNT
 C0F2 CDFBC1                  CALL    BLNK      ;SPACE OVER 8
 C0F5 F1                      POP     PSW
 C0F6 47                      MOV     B,A
 C0F7 CDE5C1                  CALL    BITS      ;PRINT TEST BYTE
 C0FA 160A                    MVI     D,0AH
 C0FC CDFBC1                  CALL    BLNK
 C0FF 7E                      MOV     A,M
 C100 CDE5C1                  CALL    BITS      ;PRINT BAD BYTE
 C103 78                      MOV     A,B       ;MOVE TEST BYTE BACK
 C104 D1                      POP     D         ;RESTORE END ADDR
 C105 C9                      RET
                      ;
                      ;
                      ;
 C106 F5              HILOX:  PUSH    A         ;SAVE ACC
 C107 23                      INX     H         ;INC CURRENT ADDR
 C108 7C                      MOV     A,H       ;LOAD HIGH ODER ADDR
 C109 BA                      CMP     D         ;COMPARE WITH END
 C10A C216C1                  JNZ     DIFF      ;NO MATCH
 C10D 7D                      MOV     A,L       ;LOAD LOW ORDER
 C10E BB                      CMP     E         ;COMPARE LOW ORDERS
 C10F C216C1                  JNZ     DIFF      ;NO MATCH
 C112 F1                      POP     A         ;MATCH END
 C113 33                      INX     SP        ;FAKE RETURN ONE LEVEL OUT
 C114 33                      INX     SP
 C115 C9                      RET
 C116 F1              DIFF:   POP     A         ;CONTINUE STUFFING
 C117 C9                      RET
 C118 0E3F            PROB:   MVI     C,3FH     ; ?
 C11A CD8EC1                  CALL    CONOT     ;PRINT ?
 C11D C304C0                  JMP     ENTRY
                      ;
                      ;
 C120 0D0A4D454DMSG1      DB      0DH,0AH,'MEMORY TEST',0
 C12E 0D0A454E54MSG2      DB      0DH,0AH,'ENTER START ADDRESS     ',0
 C149 0D0A454E54MSG3      DB      0DH,0AH,'ENTER STOP ADDRESS     ',0
 C162 0D0A454E44MSG4      DB      0DH,0AH,'END OF PASS       ',0AH,0
                      ; diagnostic input output routines
                      ; for brian donlan  26 feb 79

 0003 =               CSTAT   EQU  3          ;CONSOLE STATUS PORT.
 0003 =               CCOM    EQU  3          ;CONSOLE COMMAND PORT.
 0002 =               CDATA   EQU  2          ;CONSOLE DATA PORT.
 0002 =               CKBR    EQU  00000010B  ;KEYBOARD READY BIT.
 0001 =               CPTR    EQU  00000001B  ;PRINT READY BIT.
 0001 =               CNULL   EQU  1          ;CONSOLE NULL COUNT.

                      ; CHECK CONSOLE INPUT STATUS.
                      ;

 C177 DB03            CONST:  IN   CSTAT      ;READ CONSOLE STATUS.
 C179 E602                    ANI  CKBR       ;LOOK AT KB READY BIT.
 C17B 3E00                    MVI  A,0        ;SET A=0 FOR RETURN.
 C17D C8                      RZ              ;NOT READY WHEN ZERO.
 C17E 2F                      CMA             ;IF READY A=FF.
 C17F C9                      RET             ;RETURN FROM CONST.
```

```
                ;
                ; READ A CHARACTER FROM CONSOLE.
                ;
C180 DB03       CONIN:  IN   CSTAT           ;READ CONSOLE STATUS.
C182 E602               ANI  CKBR       ;IF NOT READY,

C184 CA80C1             JZ   CONIN      ;READY WHEN HIGH.
C187 DB02               IN   CDATA      ;READ A CHARACTER.
C189 D302               OUT    CDATA
C18B E67F               ANI  7FH        ;MAKE MOST SIG. BIT = 0.
C18D C9                 RET
                ;
                ; WRITE A CHARACTER TO THE CONSOLE DEVICE.
                ;
C18E 3E0D       CONOT:  MVI  A,0DH      ;IF IT'S A CR,
C190 B9                 CMP  C          ;THEN HOP OUT
C191 CA9FC1             JZ   CONUL      ;TO NULL ROUTINE.
C194 DB03       CONOT1: IN   CSTAT      ;READ CONSOLE STATUS.
C196 E601               ANI  CPTR       ;IF NOT READY,
C198 CA94C1             JZ   CONOT1     ;READY WHEN HIGH.
C19B 79                 MOV  A,C        ;GET CHARACTER.
C19C D302               OUT  CDATA      ;PRINT IT.
C19E C9                 RET             ;RETURN.
C19F C5         CONUL:  PUSH B          ;SAVE B&C.
C1A0 0601               MVI  B,CNULL    ;GET NULL COUNT.
C1A2 CD94C1     CONUL1: CALL CONOT1     ;PRINT CR.
C1A5 0E00               MVI  C,0        ;GET NULL CHAR.
C1A7 05                 DCR  B          ;DECREMENT COUNTER.
C1A8 C2A2C1             JNZ  CONUL1     ;DO NEXT NULL.
C1AB C1                 POP  B          ;RESTORE B&C.
C1AC 79                 MOV  A,C        ;RESTORE A.
C1AD C9                 RET             ;RETURN.
                ;

                ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                ;       PRINT MESSAGE UNTIL ZERO
                ;       MESSAGE ADDRESS REG H & L
                ;
                ;;;;;;;;;;;;;;;;;;;;;;;
C1AE 7E         PMSG:   MOV     A,M     ;GET CHAR
C1AF B7                 ORA     A       ;IS IT A  ZERO
C1B0 C8                 RZ
C1B1 4F                 MOV     C,A     ;OTHERWISE PRINT
C1B2 CD8EC1             CALL    CONOT
C1B5 23                 INX     H       ;INC ADDRESSS
C1B6 C3AEC1             JMP     PMSG
                ;
                ;
                ;
C1B9 0D0A0A4C4FMSG8     DB      0DH,0AH,0AH,'LOC.         TEST BYTE         MEMORY BYTE',0
                ;;;;;;;;;;;;;;;;;;;;;;;
                ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                ;     PRINT 8 BIT WORD IN BINARY FORMAT
                ;       INPUT:  DATA IN REG A
                ;
                ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                ;
C1E5 47         BITS:   MOV     B,A     ; DATA
C1E6 3F80               MVI     A,80H   ; MASK
C1E8 0E30       OVER:   MVI     C,30H
C1EA 5F                 MOV     E,A     ; STORE MASK
C1EB A0                 ANA     B       ; AND WITH MASK
C1EC CAF1C1             JZ      PRNT    ; JUMP IF ZERO
C1EF 0E31               MVI     C,31H
C1F1 CD8EC1     PRNT:   CALL    CONOT
C1F4 A0                 ANA     B       ; ZERO CARRY
C1F5 7B                 MOV     A,E     ; LOAD MASK
C1F6 1F                 RAR
C1F7 D2E8C1             JNC     OVER
C1FA C9                 RET
```

```
                   ;
C1FB 0E20          BLNK:   MVI     C,20H             ;PRINT BLANKS, # IN REG. D
C1FD CD94C1        LP19:   CALL    CONOT1
C200 15                    DCR     D
C201 C2FDC1                JNZ     LP19
C204 C9                    RET
                   ;
C205 7A            BINHA:  MOV     A,D
C206 1F                    RAR
C207 1F                    RAR
C208 1F                    RAR
C209 1F                    RAR
C20A CD22C2                CALL    BIN1
C20D 4F                    MOV     C,A
C20E CD8EC1                CALL    CONOT
C211 7A                    MOV     A,D
C212 CD22C2                CALL    BIN1
C215 4F                    MOV     C,A
C216 CD8EC1                CALL    CONOT
C219 C9                    RET
                   ;
                   ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                   ;    OUTPUTS FOUR HEX DIGITS IN ASCII
                   ;    ENTER WITH DATA IN REG PAIR E AND D
                   ;;;;;;;;;;;;;;;;;
                   ;;
                   ;
C21A CD05C2        BINB:   CALL    BINHA
C21D 53                    MOV     D,E
C21E CD05C2                CALL    BINHA
C221 C9                    RET
                   ;
                   ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                   ;    CONVERTS HEX TO ASCII
                   ;    INPUT: 4 BITS HEX  REG A
                   ;    OUTPUT:  8 BIT ASSCII REG A
                   ;
                   ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                   ;;;
                   ;
C222 E60F          BIN1:   ANI     0FH
C224 C630                  ADI     30H
C226 FE3A                  CPI     3AH
C228 D8                    RC
C229 C607                  ADI     07H
C22B C9                    RET
                   ;
                   ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                   ;    INPUTS 4 DIGITS FROM CONSOLE
                   ;    RETURN;  4 HEX DIGITS IN REG E-D
                   ;
                   ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                   ;;
                   ;
C22C CD80C1        BBIN:   CALL    CONIN
C22F CD5BC2                CALL    AHS1
C232 17                    RAL
C233 17                    RAL
C234 17                    RAL
C235 17                    RAL
C236 E6F0                  ANI     0F0H
C238 57                    MOV     D,A
C239 CD80C1                CALL    CONIN
C23C CD5BC2                CALL    AHS1
C23F E60F                  ANI     0FH
C241 B2                    ORA     D
C242 57                    MOV     D,A
C243 CD80C1                CALL    CONIN
C246 CD5BC2                CALL    AHS1
C249 17                    RAL
C24A 17                    RAL
C24B 17                    RAL
```

```
C24C 17                 RAL
C24D E6F0               ANI     0F0H
C24F 5F                 MOV     E,A
C250 CD80C1             CALL    CONIN
C253 CD5BC2             CALL    AHS1
C256 E60F               ANI     0FH
C258 B3                 ORA     E
C259 5F                 MOV     E,A
C25A C9                 RET
                ;
                ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                ;     CONVERT ASCII TO HEX
                ;     INPUT;  8 BIT ASCII  REG A
                ;     OUTPUT:  4 BIT HEX REG A
                ;
                ;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                ;;
                ;
C25B 00         AHS1:   NOP
C25C D630               SUI     30H
C25E FE0A               CPI     0AH
C260 D8                 RC
C261 D607               SUI     07H
C263 C9                 RET
                ;
                ;;;;;;;;;;
                ;
                ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                .       ....ATE SIO PORTS
                ;
                ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                ;;
                ;
C264 3EAA       INITA:  MVI     A,0AAH          ;GET DUMMY MODE WORD
C266 D303                OUT    CSTAT           ;OUTPUT IT
C268 3E40               MVI     A,40H           ;GET RESET BIT
C26A D303               OUT     CSTAT           ;RESET SIO BOARD
C26C 3ECE               MVI     A,0CEH          ;GET REAL MODE WORD
C26E D303               OUT     CSTAT           ;SET THE MODE FOR REAL
C270 3E37               MVI     A,37H           ;GET THE COMMAND
C272 D303               OUT     CSTAT           ;OUTPUT IT
C274 C9                 RET
                ;
                ;
                ;
C275 0E0D       CRLF:   MVI     C,13            ;CR
C277 CD8EC1             CALL    CONOT
C27A 0E0A       LF:     MVI     C,10            ;LF
C27C CD94C1             CALL    CONOT1
C27F 0E7F               MVI     C,7FH
C281 CD94C1             CALL    CONOT1
C284 CD8EC1             CALL    CONOT
C287 C9                 RET
```

```
                    ;
                    ;
                    ;  MINI-MEMORY TEST
                    ;PROM VERSION FOR  0 TO 100H
                    ;
                    ;  BRIAN J. DONLAN
                    ;
                    ;
C290                          ORG       0C290H
C290 F3             ENTER:    DI
C291 3EFE                     MVI       A,0FEH
C293 D3FF                     OUT       0FFH               ;OUTPUT PHASE I LITES
C295 210000                   LXI       H,000H             ;START ADDRESS
C298 AF             LP2:      XRA       A                  ;ZERO ACC
C299 77             LP1:      MOV       M,A                ;STORE TEST PATTERN IN MEM.
C29A 46                       MOV       B,M                ;READ BACK TO B
C29B B8                       CMP       B                  ;COMPARE FOR OK
C29C C2FDC2                   JNZ       ERR1               ;JUMP IF ERROR
C29F 3C                       INR       A                  ;NEW TEST PATTERN
C2A0 C299C2                   JNZ       LP1
C2A3 23                       INX       H
C2A4 1100FF                   LXI       D,0FF00H                  ;STOP ADDRESS
C2A7 EB                       XCHG
C2A8 19                       DAD       D                  ;ADD TWO'S COMPLIMENT
C2A9 EB                       XCHG
C2AA D298C2                   JNC       LP2
                    ;
                    ; PHASE II
C2AD 3EFD                     MVI       A,0FDH             ;PHASE II LITES
C2AF D3FF                     OUT       0FFH
C2B1 210000                   LXI       H,000H
C2B4 74             LP3:      MOV       M,H                ;LOW ADDRESS TO MEM
C2B5 23                       INX       H
C2B6 1100FF                   LXI       D,0FF00H           ;STOP ADDRESS
C2B9 EB                       XCHG
C2BA 19                       DAD       D
C2BB EB                       XCHG
C2BC D2B4C2                   JNC       LP3
                    ;         READ MEMORY
C2BF 210000                   LXI       H,000H
C2C2 7E             LP4:      MOV       A,M                ;READ MEMORY
C2C3 94                       SUB       H                  ;COMPARE
C2C4 C223C3                   JNZ       ERR2               ;JUMP IF ERROR
C2C7 23                       INX       H
C2C8 1100FF                   LXI       D,0FF00H
C2CB EB                       XCHG
C2CC 19                       DAD       D
C2CD EB                       XCHG
C2CE D2C2C2                   JNC       LP4
                    ;
                    ; PHASE III
                    ;
C2D1 3EFC                     MVI       A,0FCH
C2D3 D3FF                     OUT       0FFH               ;PHASE THREE LITES
C2D5 210000                   LXI       H,000H
C2D8 75             LP5:      MOV       M,L                ;STORE HIGH ADDRESS IN ALL  MEM
C2D9 23                       INX       H
C2DA 1100FF                   LXI       D,0FF00H
C2DD EB                       XCHG
C2DE 19                       DAD       D
C2DF EB                       XCHG
C2E0 D2D8C2                   JNC       LP5
                    ;READ MEM
C2E3 210000                   LXI       H,000H
C2E6 7E             LP6:      MOV       A,M                ;READ MEMORY
C2E7 95                       SUB       L                  ;COMPARE
C2E8 C22FC3                   JNZ       ERR3
C2EB 23                       INX       H
C2EC 1100FF                   LXI       D,0FF00H
C2EF EB                       XCHG
C2F0 19                       DAD       D
C2F1 EB                       XCHG
C2F2 D2E6C2                   JNC       LP6
```

```
                    ;
                    ;
                    ;   ALL PHASE COMPLETE
C2F5 3EFF                   MVI     A,OFFH
C2F7 2190C2                 LXI     H,ENTER
C2FA C33BC3                 JMP     LITES              ;GO TO LITES PROG
                    ;
                    ;
                    ;PHASE I ERROR
C2FD EB             ERR1:   XCHG
C2FE 4F                     MOV     C,A                ;SAVE BAD DATA
C2FF 2107C3                 LXI     H,COMERR                ;RETURN
C302 3EF1                   MVI     A,OF1H             ;PHASE I ERROR LITES
C304 C33BC3                 JMP     LITES
                    ;
                    ;
                    ;COMMON ERROR OUTPUT ROUTINE
C307 7A             COMERR: MOV     A,D                ;HIGH ADDRESS
C308 210EC3                 LXI     H,LOADD            ;RETURN
C30B C33BC3                 JMP     LITES
C30E 7B             LOADD:  MOV     A,E                ;LOW ADDRES TO LITES
C30F 2115C3                 LXI     H,TPAT             ;RETURN
C312 C33BC3                 JMP     LITES
C315 79             TPAT:   MOV     A,C                ;TEST PATTERN TO LITES
C316 211CC3                 LXI     H,ACTDAT           ;RETURN
C319 C33BC3                 JMP     LITES
-31C 78             ACTDAT: MOV     A,B                ;ACTUAL DATA TO LITES
C31D 2190C2                 LXI     H,ENTER                 ;START OVER
C320 C33BC3                 JMP     LITES
                    ;;
                    ;
                    ; PHASE II ERROR
C323 EB             ERR2:   XCHG                       ;SAVE BAD ADDRESS
C324 82                     ADD     D
C325 47                     MOV     B,A
C326 4A                     MOV     C,D
C327 3EF2                   MVI     A,OF2H             ;PHASE II ERROR TO LITES
C329 2107C3                 LXI     H,COMERR                ;RETURN
C32C C33BC3                 JMP     LITES
                    ;
                    ;
                    ; PHASE III ERROR
C32F EB             ERR3:   XCHG            ;SAVE BAD ADDRESS
C330 83                     ADD     E
C331 47                     MOV     B,A
C332 4B                     MOV     C,E
C333 3EF3                   MVI     A,OF3H             ;PHASE II ERRO TO LITES
C335 2107C3                 LXI     H,COMERR           ;RETURN
C338 C33BC3                 JMP     LITES
                    ;
                    ;
                    ;LITES ROUTINE    ENTER WITH RETURN IN REG H&L
                    ;                            DATA FOR LITES IN A
C33B 2F             LITES:  CMA
C33C D3FF                   OUT     OFFH               ;OUTPUT LITES
C33E F9                     SPHL                       ;SAVE RETURN IN SP
C33F DBFF                   IN      OFFH               ;READ SENSE SWITCHES
C341 67                     MOV     H,A                ;SAVE IN H
C342 DBFF           LP7:    IN      OFFH               ;READ SWITCHES
C344 AC                     XRA     H                  ;SEE IF THEY CHANGED
C345 CA42C3                 JZ      LP7
C348 2118FC                 LXI     H,OFC18H                ;DELAY LOOP
C34B 23             LP8:    INX     H
C34C AF                     XRA     A
C34D B4                     ORA     H
C34E C24BC3                 JNZ     LP8
C351 210000                 LXI     H,0                ;ZERO H
C354 39                     DAD     SP                 ;MOVE RETURN BACK TO H & L
C355 E9                     PCHL                       ;RETURN
```

```
                  ;       DISK TEST FOR TARBELL DISK CONTROLLER
                  ;       BRIAN J. DONLAN
                  ;       18  MAR  79
                  ;    PROM VERSION
                  ;
                  ;
C800                        ORG     0C800H
C800 217000       ENTRY1:   LXI     H,070H          ;SET STACK POINTER
C803 F9                     SPHL
C804 F3                     DI
C805 CD52CD       ENTRY:    CALL    INITA           ;INITI TTY
C808 2162C9                 LXI     H,MSG1          ;OPENING MESSAGE
C80B CD8CCC                 CALL    PMSG
C80E 2185C9                 LXI     H,MSG1A
C811 CD8CCC                 CALL    PMSG
C814 CDDACA                 CALL    CONIN           ;CHECK KEYBOARD
C817 FE59                   CPI     'Y'             ;CHECK IF Y
C819 C205C8                 JNZ     ENTRY           ;  ?? START OVER
C81C CD63CD                 CALL    CRLF
C81F CD63CD       LOOP6:    CALL    CRLF
C822 AF                     XRA     A               ;ZERO ACC
C823 320800                 STA     ERRFLG          ;ZERO EERROR FLAG
C826 32040D                 STA     LPCNT           ;ZERO LOOP COUNT
C829 CD08CB                 CALL    HOME            ;HOME DRIVE TO TRK 0
C82C AF           LOOP4:    XRA     A
C82D 320500                 STA     INNER           ;ZERO INNER TRK
C830 3E26                   MVI     A,38            ;OUTER TRK
C832 320600                 STA     OUTER
C835 CD82C8                 CALL    PAT             ;GET PATTERN
C838 CDB5C8                 CALL    INWRT
C83B 3E22                   MVI     A,34
C83D CD72CC                 CALL    SEEK            ;
C840 CDE7C8                 CALL    INRD            ;MOVE BACK AND CHECK TRK 00
C843 3E01                   MVI     A,01            ;SET UP TO DO PAIRS
C845 320500                 STA     INNER           ;START PAIRS WITH TRK01
                  ;
                  ; TEST FOR CONSOLE INTERRUPT
C848 DB03         LOOP8:    IN      CSTAT
C84A E602                   ANI     02H             ;KEYBAORD READY
C84C CA5BC8                 JZ      LOOP3           ;NO
C84F DB02                   IN      CDATA           ;READ KEYS
C851 FE03                   CPI     03H             ;CONTROL C
C853 CA00F0                 JZ      RENT            ;RETURN TO MONITOR
C856 FE02                   CPI     02H             ;CONTROLB
C858 CA05C8                 JZ      ENTRY           ;START OVER AGAIN
                            ;
C85B CDB5C8       LOOP3:    CALL    INWRT           ;WRITE INNER TRK
C85E CDDEC8                 CALL    OUTWRT
C861 CDE7C8                 CALL    INRD            ;READ INNER TRK
C864 CD17C9                 CALL    OUTRD
C867 3A0500                 LDA     INNER
C86A 3C                     INR     A
C86B 320500                 STA     INNER
C86E C626                   ADI     38              ;FIND NEXT OUTER T4K
C870 320600                 STA     OUTER           ;STORE OUTER TRK
C873 FE4D                   CPI     77              ;TRK    77 YET ?
C875 C248C8                 JNZ     LOOP8           ;NOT DONE YET
C878 3A0400                 LDA     LPCNT           ;LOOP COUNTER
C87B 3C                     INR     A
C87C 320400                 STA     LPCNT
C87F C32CC8                 JMP     LOOP4
```

```
                    ;
                    ;       PATTERN ROUNTINE EXPANDABLE
C882 3A0400         PAT:    LDA     LPCNT           ;LOAD LOOP COUNTER
C885 CAA3C8                 JZ      IST
C888 FE01                   CPI     01              ;SECOND PASS
C88A CAA9C8                 JZ      SECD
C88D FE02                   CPI     02
C88F CAAFC8                 JZ      THIRD
C892 21ACC9                 LXI     H,MSG2          ;END OF PASS
C895 CD8CCC                 CALL    PMSG
C898 DB03                   IN      CSTAT           ;CHECK KEYBOARD
C89A E602                   ANI     02H
C89C CA1FC8                 JZ      LOOP6           ;CONTINUE TEST UNTIL INTERUPTED
C89F 76                     HLT
C8A0 C305C8                 JMP     ENTRY
C8A3 3EFF           IST:    MVI     A,OFFH          ;ALL ONES PATERN
C8A5 320700                 STA     PATEN           ;STORE PATTERN
C8A8 C9                     RET
C8A9 3E00           SECD:   MVI     A,OOH           ;ALL ZERO PATTERN
C8AB 320700                 STA     PATEN
C8AE C9                     RET
C8AF 3E55           THIRD   MVI     A,55H           ;ALTER PATTERN
C8B1 320700                 STA     PATEN
C8B4 C9                     RET
                    ;
                    ;
                    ;       WRITE INNER TRK
C8B5 3A0500         INWRT:  LDA     INNER
C8B8 320B00                 STA     TRK
C8BB CD72CC         BOTH:   CALL    SEEK            ;MOVE HEAD TO TRK
C8BE 3E01                   MVI     A,01            ;FIRST SECTOR
C8C0 320C00                 STA     SECT
                    ;
C8C3 AF             LOOP1:  XRA     A               ;ZERO ACC
C8C4 320D00                 STA     REPETE          ;ZERO REPEAT FLAG
C8C7 CD37CC                 CALL    WRITE           ;WRITE ONE SECTOR
C8CA 3A0D00                 LDA     REPETE          ;LOAD REPEAT FLAG
C8CD B7                     ORA     A               ;SET FLAGS
C8CE C2C3C8                 JNZ     LOOP1           ;REPEAT SECTOR
C8D1 3A0C00                 LDA     SECT
C8D4 3C                     INR     A               ;INC SECTOR
C8D5 320C00                 STA     SECT
C8D8 FE1B                   CPI     27              ;ALL SECTOR DONE ?
C8DA C2C3C8                 JNZ     LOOP1           ;NO
C8DD C9                     RET
                    ;
                    ;
                    ;       WRITE OUTER TRK
C8DE 3A0600         OUTWRT: LDA     OUTER           ;LOAD OUTER TRK
C8E1 320B00                 STA     TRK
C8E4 C3BBC8                 JMP     BOTH            ;COMMON WRITE ROUNTINE
                    ;
                    ;
                    ;       READ INNER TRK
C8E7 3A0500         INRD:   LDA     INNER
C8EA 320B00                 STA     TRK
C8ED CD72CC         BOTH2:  CALL    SEEK            ;MOVE HEAD TO TRK
C8F0 3E01                   MVI     A,01            ;FIRST SECTOR
C8F2 320C00                 STA     SECT            ;ZERO SECTOR
                    ;
C8F5 AF             LOOP5:  XRA     A
C8F6 320800                 STA     ERRFLG          ;ZREO ERROR COUNT
C8F9 320D00                 STA     REPETE
C8FC CD3BCB                 CALL    READ            ;READ ONE SECTOR
C8FF AF                     XRA     A               ;ZERO ACC
C900 320800                 STA     ERRFLG
C903 3A0D00                 LDA     REPETE          ;REPEAT FLAG
C906 B7                     ORA     A               ;SET FLAGS
C907 C2F5C8                 JNZ     LOOP5
C90A 3A0C00                 LDA     SECT
C90D 3C                     INR     A
C90E 320C00                 STA     SECT            ;NEXT SECTOR
C911 FE1B                   CPI     27              ;ALL SECTORS DONE ?
C913 C2F5C8                 JNZ     LOOP5           ;NO
C916 C9                     RET
```

```
                  ;
                  ;       READ OUTER TRK
C917 3A0600       OUTRD:  LDA     OUTER              ;OUTER TRK NO.
C91A 320B00               STA     TRK
C91D C3EDC8               JMP     BOTH2
                  ;
C920 21BCC9       ERRPNT: LXI     H,MSG3             ;ERROR MESSAGE
C923 CD8CCC               CALL    PMSG
C926 3A0800               LDA     ERRFLG             ;ERROR COUNT
C929 57                   MOV     D,A
C92A CD2BCD               CALL    BINHA              ;PRINT  ERROR COUNT
C92D 21F3C9               LXI     H,MSG4             ;HEADINGS
C930 CD8CCC               CALL    PMSG
C933 1603                 MVI     D,03               ;SPACE OVER
C935 CD21CD               CALL    BLNK
C938 3A0B00               LDA     TRK                ;TRACK NO.
C93B 57                   MOV     D,A
C93C CD2BCD               CALL    BINHA              ;PRINT TRACK NO.
C93F 1610                 MVI     D,16               ;SPACE OVER
C941 CD21CD               CALL    BLNK
C944 3A0C00               LDA     SECT               ;SECTOR NO.
C947 57                   MOV     D,A
C948 CD2BCD               CALL    BINHA              ;PRINT SECTOR NO.
C94B 160D                 MVI     D,13               ;SPACE OVER
C94D CD21CD               CALL    BLNK
C950 3A0700               LDA     PATEN
C953 CD0BCD               CALL    BITS               ;PRINT TEST PATTERN
C956 160C                 MVI     D,12               ;SPACE OVE
C958 CD21CD               CALL    BLNK
C95B 3A0900               LDA     BADBT              ;LAST BAD BYTE
C95E CD0BCD               CALL    BITS               ;PRINT LAST BAD BYTE
C961 C9                   RET
                  ;
0004 =            LPCNT:  EQU     4                  ;SPACE FOR LOOP COUNTER
0005 =            INNER:  EQU     5                  ;SPACE FOR INNER TRK NO.
0006 =            OUTER:  EQU     6                  ;SPACE FOR OUTER TRK NO.
0007 =            PATEN   EQU     7                  ;SPACE FOR TEST PATTERN
0008 =            ERRFLG: EQU     8                  ;SPACE FOR ERROR COUNT
0009 =            BADBT:  EQU     9                  ;SPACE FOR BAD BYTE
000A =            BDTRK:  EQU     0AH                ;SPACE FOR DISK READ TRK WHEN ERR
000D =            REPETE: EQU     0DH                ;REPETE FLAG
C962 0D0A444953MSG1:      DB      0DH,0AH,'DISK TEST NO. 1 FORMATTED TEST  ',0
C985 0D0A4C4F41MSG1A:     DB      0DH,0AH,'LOAD SCRATCH DISK  TYPE Y WHEN READY',0
C9AC 0D0A20454EMSG2       DB      0DH,0AH,' END OF PASS ',0
C9BC 0D0A444154MSG3:      DB      0DH,0AH,'DATA ERROR ON DISK CHECK       ERROR COUNT IN HEX   ',0
C9F3 0D0A205452MSG4:      DB      0DH,0AH,' TRACK NO.          SECTOR NO.        TEST BYTE          LAST ERROR'
CA37 0D0A484541MSG5:      DB      0DH,0AH,'HEAD POSITION ',0
CA48 0D0A444953MSG6:      DB      0DH,0AH,'DISK TRACK              CONTROLLER TRACK           SECTOR ',0DH,0AH,
CA83 0D0A0D0A20MSG7:      DB      0DH,0AH,0DH,0AH,'  !! EXECUTION STOPPED !! ',0
CAA2 0D0A545950MSG8:      DB      0DH,0AH,'TYPE R TO RETRY,  C TO CONTINUE,  ANYTHING ELSE STOP ',0
                  ;
0003 =            CSTAT   EQU  3           ;CONSOLE STATUS PORT.
0003 =            CCOM    EQU  3           ;CONSOLE COMMAND PORT.
0002 =            CDATA   EQU  2           ;CONSOLE DATA PORT.
0002 =            CKBR    EQU  00000010B   ;KEYBOARD READY BIT.
0001 =            CPTR    EQU  00000001B   ;PRINT READY BIT.
0001 =            CNULL   EQU  1           ;CONSOLE NULL COUNT.
00F8 =            DISK    EQU  0F8H        ;DISK BASE ADDRESS.
00F8 =            DCOM    EQU  DISK        ;DISK COMMAND PORT.
00F8 =            DSTAT   EQU  DISK        ;DISK STATUS PORT.
00F9 =            TRACK   EQU  DISK+1      ;DISK TRACK PORT.
00FA =            SECTP   EQU  DISK+2      ;DISK SECTOR PORT.
00FB =            DDATA   EQU  DISK+3      ;DISK DATA PORT.
00FC =            WAIT    EQU  DISK+4      ;DISK WAIT PORT.
00FC =            DCONT   EQU  DISK+4      ;DISK CONTROL PORT.

000B =            TRK:    EQU     0BH                ;ADDRESS FOR TRACK
000C =            SECT:   EQU     0CH                ;ADDRESS FOR SECTOR
```

```
                   ;
                   ; READ A CHARACTER FROM CONSOLE.
                   ;
CADA DB03          CONIN:   IN    CSTAT              ;READ CONSOLE STATUS.
CADC E602                   ANI   CKBR          ;IF NOT READY,

CADE CADACA                 JZ    CONIN         ;READY WHEN HIGH.
CAE1 DB02                   IN    CDATA         ;READ A CHARACTER.
CAE3 D302                   OUT     CDATA
CAE5 E67F                   ANI   7FH           ;MAKE MOST SIG. BIT = 0.
CAE7 C9                     RET
                   ;
                   ; WRITE A CHARACTER TO THE CONSOLE DEVICE.
                   ;
CAE8 3E0D          CONOT:   MVI   A,0DH         ;IF IT'S A CR,
CAEA B9                     CMP   C             ;THEN HOP OUT
CAEB CAF9CA                 JZ    CONUL         ;TO NULL ROUTINE.
CAEE DB03          CONOT1:  IN    CSTAT         ;READ CONSOLE STATUS.
CAF0 E601                   ANI   CPTR          ;IF NOT READY,
CAF2 CAEECA                 JZ    CONOT1        ;READY WHEN HIGH.
CAF5 79                     MOV   A,C           ;GET CHARACTER.
CAF6 D302                   OUT   CDATA         ;PRINT IT.
CAF8 C9                     RET                 ;RETURN.
CAF9 C5            CONUL:   PUSH  B             ;SAVE B&C.
CAFA 0601                   MVI   B,CNULL       ;GET NULL COUNT.
CAFC CDEECA        CONUL1:  CALL  CONOT1        ;PRINT CR.
CAFF 0E00                   MVI   C,0           ;GET NULL CHAR.
CB01 05                     DCR   B             ;DECREMENT COUNTER.
CB02 C2FCCA                 JNZ   CONUL1        ;DO NEXT NULL.
CB05 C1                     POP   B             ;RESTORE B&C.
CB06 79                     MOV   A,C           ;RESTORE A.
CB07 C9                     RET                 ;RETURN.
                   ;
                   ; MOVE DISK TO TRACK ZERO.
                   ;
CB08 3ED0          HOME:    MVI   A,0D0H        ;CLEAR ANY PENDING COMMAND.
CB0A D3F8                   OUT   DCOM
CB0C AF                     XRA      A          ;ZERO ACC
CB0D 320B00                 STA      TRK        ;STORE TRACK
CB10 DBF8          HOME1:   IN    DSTAT         ;READ DISK STATUS.
CB12 0F                     RRC                 ;LOOK AT LSB.
CB13 DA10CB                 JC    HOME1         ;WAIT FOR NOT BUSY.
CB16 3E03                   MVI   A,3           ;20 MS STEP RATE.
CB18 D3F8                   OUT   DCOM          ;ISSUE HOME COMMAND.
CB1A DBFC                   IN    WAIT          ;WAIT FOR INTRQ.
CB1C B7                     ORA   A             ;SET FLAGS.
CB1D FA2CCB                 JM    HERR          ;ERROR IF DRQ.
CB20 DBF8                   IN    DSTAT         ;READ DISK STATUS.
CB22 57                     MOV   D,A           ;SAVE IN REGISTER D.
CB23 E604                   ANI   4             ;LOOK AT BIT 2.
CB25 CA2CCB                 JZ    HERR          ;ERROR IF NOT TRK 0.
CB28 7A                     MOV   A,D           ;GET STATUS BACK.
CB29 E691                   ANI   91H           ;MASK NON-ERROR BITS.
CB2B C8                     RZ                  ;RETURN IF NO ERROR.
CB2C 21FACC        HERR:    LXI   H,HEMSG       ;PRINT "HOME ".
CB2F 7A                     MOV   A,D           ;MASK NON-ERROR BITS.
CB30 E691                   ANI   91H
CB32 57                     MOV   D,A
CB33 C374CB                 JMP   ERMSG         ;DO COMMON ERROR MSGS.
                   ;
                   ; SELECT DISK NUMBER.
                   ;
CB36 3E02          INTDSK: MVI      A,02             ;DRIVE NO. 1
CB38 D3FC          DSK1:    OUT   DCONT         ;SET THE LATCH WITH CODE.
CB3A C9                     RET                 ;RETURN FROM SELDSK.
```

```
               ;
               ; READ THE SECTOR AT SECT, FROM THE PRESENT TRACK.
               ;  SECTOR IN  SECT
               ;   HEAD LOAD FIRST
               ;
CB3B 218000    READ:   LXI     H,080H          ;READ BUFFER
CB3E 3A0C00            LDA     SECT
CB41 D3FA      READ1:  OUT  SECTP      ;SET SECTOR INTO 1771.
CB43 3E8C              MVI  A,8CH      ;CODE FOR READ W/O HD LD.
CB45 D3F8      READE:  OUT  DCOM       ;SEND COMMAND TO 1771.
CB47 DBFC      RLOOP:  IN   WAIT       ;WAIT FOR DRQ OR INTRQ.
CB49 B7                ORA  A          ;SET FLAGS.
CB4A F254CB            JP   RDDONE     ;DONE IF INTRQ.
CB4D DBFB              IN   DDATA      ;READ A DATA BYTE FROM DISK.
               ;
CB4F 77                MOV     M,A             ;STORE IN BUFFER
CB50 23                INX     H               ;INC BUFF POINTER
               ;
CB51 C347CB            JMP     RLOOP
               ;
               ; COMPARE DATA WITH TEST BYTE;
CB54 218000    RDDONE: LXI     H,080H          ;HEAD OF BUFFER
CB57 3A0700            LDA     PATEN           ;TEST PATTERN
CB5A 47                MOV     B,A             ;PATTERN TO B
CB5B 1680              MVI     D,080H          ;COUNTER FOR BYTES
CB5D 7E        COMPLP: MOV     A,M             ;GET DATA
CB5E B8                CMP     B               ;COMPARE WITH TB
CB5F C22ACC            JNZ     DATERR          ;ERROR
CB62 23        ERRET:  INX     H
CB63 15                DCR     D               ;DEC BYTE COUNT
CB64 C25DCB            JNZ     COMPLP          ;DO 128 TIMES
CB67 DBF8              IN   DSTAT      ;READ DISK STATUS.
CB69 E69D              ANI  9DH        ;LOOK AT ERROR BITS.
CB6B 57                MOV     D,A     ;SAVE ERROR BITS
CB6C 3A0800            LDA     ERRFLG          ;READ ERROR FLAG
CB6F B2                ORA     D               ;SET FLAGS ON COMBO
CB70 C8                RZ              ;RETURN IF NONE.
CB71 21E1CC            LXI  H,RDMSG    ;PRINT "READ ".
CB74 CD8CCC    ERMSG:  CALL PMSG       ;PRINT ORIGIN MESSAGE.
               ;
               ;
               ;       COMMON ERROR PRINT OUT
               ;

CB77 7A        ERMSG1: MOV  A,D        ;GET ERROR BITS.
CB78 E680              ANI  80H        ;IF BIT 7 HIGH,
CB7A 2197CC            LXI  H,NRMSG    ;"NOT READY".
CB7D C48CCC            CNZ PMSG
CB80 7A                MOV  A,D        ;GET ERROR BITS.
CB81 E610              ANI  10H        ;IF BIT 4 IS HIGH,
CB83 21A2CC            LXI  H,RNMSG    ;PRINT "RECORD NOT FOUND"
CB86 C48CCC            CNZ PMSG
CB89 7A                MOV  A,D        ;GET ERROR BITS.
CB8A E608              ANI  8H         ;IF BIT 3 IS HIGH,
CB8C 21B4CC            LXI  H,CRCMSG   ;PRINT "CRC ERROR".
CB8F C48CCC            CNZ PMSG
CB92 7A                MOV  A,D        ;GET ERROR BITS.
CB93 E604              ANI  4H         ;IF BIT 2 IS HIGH,
CB95 21B9CC            LXI  H,LDMSG    ;PRINT "LOST DATA".
CB98 C48CCC            CNZ PMSG
CB9B 7A                MOV  A,D        ;GET ERROR BITS.
CB9C E601              ANI  1          ;IF BIT 1 IS HIGH,
CB9E 21C4CC            LXI  H,BSYMSG   ;PRINT "BUSY".
CBA1 C48CCC            CNZ  PMSG
CBA4 21DACC    PERMSG: LXI  H,ERRMSG   ;PRINT "ERROR."
CBA7 CD8CCC            CALL PMSG
CBAA 7A                MOV     A,D             ;MOVE FLAGS TO ACC
CBAB E618              ANI     18H             ;CRC OR RECORD NOT FOUND
CBAD CAEFCB            JZ      RETRY
CBB0 3EC4      TRKCHK: MVI     A,0C4H
CBB2 D3F8              OUT     DCOM            ;READ ADDRESS
CBB4 DBFC              IN      WAIT
```

```
CBB6 DBFB                IN      DDATA           ;TRACK ADDRESS
CBB8 320A00              STA     BDTRK
CBBB DBFC       CHKS2    IN      WAIT            ;DUMP REST OF DATA
CBBD FABBCB              JM      CHKS2
CBC0 2137CA              LXI     H,MSG5          ;HEAD ERROR MESSAGE
CBC3 CD8CCC              CALL    PMSG
CBC6 2148CA              LXI     H,MSG6          ;HEADINGS
CBC9 CD8CCC              CALL    PMSG
CBCC 1605                MVI     D,05H
CBCE CD21CD              CALL    BLNK            ;SPACE OVER
CBD1 3A0A00              LDA     BDTRK           ;DISK TRK
CBD4 57                  MOV     D,A
CBD5 CD2BCD              CALL    BINHA           ;PRINT TRK
CBD8 1615                MVI     D,15H
CBDA CD21CD              CALL    BLNK            ;SPACE OVER
CBDD DBF9                IN      TRACK
CBDF 57                  MOV     D,A
CBE0 CD2BCD              CALL    BINHA           ;PRINT TRK
CBE3 1613                MVI     D,13H
CBE5 CD21CD              CALL    BLNK
CBE8 3A0C00              LDA     SECT            ;SECTOR
CBEB 57                  MOV     D,A
CBEC CD2BCD              CALL    BINHA           ;PRINT SECTO NO.
CBEF 3A0800     RETRY:   LDA     ERRFLG
CBF2 B7                  ORA     A               ;SET FLAGS
CBF3 C420C9              CNZ     ERRPNT          ;GO TO READ CHECK ERROR PRINT
CBF6 DB02                IN      CDATA           ;CLEAR KEYBOARD
CBF8 DBFF                IN      OFFH            ;READ SENSE SWITCHES
CBFA E601                ANI     01H             ;SWITCH 0
CBFC C221CC              JNZ     CONT
CBFF 21A2CA              LXI     H,MSG8
CC02 CD8CCC              CALL    PMSG            ;REQUEST INPUT
CC05 CDDACA              CALL    CONIN           ;READ KEYS
CC08 FE52                CPI     'R'             ;CHECK FOR R
CC0A CA15CC              JZ      FIX
CC0D FE43                CPI     'C'             ;CHECK FOR C
CC0F CA21CC              JZ      CONT
CC12 C300F0              JMP     RENT
CC15 3E01       FIX:     MVI     A,01            ;SET REPETE FLAG
CC17 320D00              STA     REPETE
CC1A CD63CD              CALL    CRLF
CC1D CD63CD              CALL    CRLF
CC20 C9                  RET
CC21 CD63CD     CONT:    CALL    CRLF
CC24 CD63CD              CALL    CRLF
CC27 3E01                MVI     A,01
CC29 C9                  RET
                ;
CC2A 320900     DATERR:  STA     BADBT           ;SAVE BAD BYTE
CC2D 3A0800              LDA     ERRFLG          ;LOAD ERROR COUNT
CC30 3C                  INR     A
CC31 320800              STA     ERRFLG          ;NEW COUNT
CC34 C362CB              JMP     ERRET           ;RETURN
                ;
                ;
                ; WRITE THE SECTOR AT SECT, ON THE PRESENT TRACK.
                ; USE STARTING ADDRESS AT DMAADD.
                ;  LOAD HEAD FIRST
                ;
CC37 3A0700     WRITE:   LDA     PATEN
CC3A 47                  MOV     B,A     ;TEST PATTERN IN B
CC3B 3A0C00              LDA     SECT    ;LOAD SECTOR
CC3E D3FA       WRITE1:  OUT  SECTP      ;SET THE SECTOR INTO 1771.
CC40 3EAC                MVI  A,0ACH     ;SET UP 1771 FOR WRITE.
CC42 D3F8                OUT  DCOM
CC44 DBFC       WLOOP:   IN   WAIT       ;WAIT FOR READY.
CC46 B7                  ORA  A          ;SET FLAGS.
CC47 F251CC              JP   WDONE      ;HOP OUT WHEN DONE.
                ;
                ;      INSERT PATTERN HERE
CC4A 78                  MOV     A,B     ;LOAD TEST PATTERN
```

```
CC4B D3FB               OUT  DDATA      ;WRITE ONTO DISK.
CC4D 23                 INX  H          ;INCREMENT MEM PTR.
CC4E C344CC             JMP  WLOOP      ;KEEP WRITING.
CC51 DBF8      WDONE:   IN   DSTAT      ;READ DISK STATUS.
CC53 E6FD               ANI  OFDH       ;LOOK AT THESE BITS.
CC55 57                 MOV     D,A     ;SAVE STATUS BITS
CC56 C8        PROCER:  RZ              ;RETURN IF NO ERR.
CC57 21E9CC    WERRO:   LXI  H,WTMSG    ;PRINT "WRITE ".
CC5A CD8CCC             CALL PMSG
CC5D 7A                 MOV  A,D        ;GET ERROR BITS.
CC5E E640               ANI  40H        ;LOOK AT BIT 6.
CC60 21CACC             LXI  H,WPMSG    ;PRINT "PROTECT ".
CC63 C48CCC             CNZ  PMSG
CC66 7A                 MOV  A,D        ;GET ERROR BITS.
CC67 E620               ANI  20H        ;LOOK AT BIT 5.
CC69 21D3CC             LXI  H,WFMSG    ;PRINT "FAULT ".
CC6C C48CCC             CNZ  PMSG
CC6F C377CB             JMP  ERMSG1     ;DO COMMON MESSAGES.
               ;
               ; MOVE THE HEAD TO THE TRACK IN REGISTER A.
               ;
CC72 D3FB      SEEK:    OUT  DDATA      ;TRACK TO DATA REGISTER.
CC74 DBF8      BUSY:    IN   DSTAT      ;READ DISK STATUS.
CC76 OF                 RRC             ;LOOK AT BIT 0.
CC77 DA74CC             JC   BUSY       ;WAIT TILL NOT BUSY.
CC7A 3E12               MVI  A,12H      ;SET FOR 10 MS STEP.
CC7C D3F8               OUT  DCOM       ;ISSUE SEEK COMMAND.
CC7E DBFC               IN   WAIT       ;WAIT FOR INTRQ.
CC80 DBF8               IN   DSTAT      ;READ STATUS.
CC82 E691               ANI  91H        ;LOOK AT BITS.
CC84 57                 MOV     D,A     ; SAVE STATUS
CC85 C8                 RZ              ;RETURN IF NO ERROR
CC86 21F2CC             LXI  H,SKMSG    ;PRINT "SEEK ".
CC89 C374CB             JMP  ERMSG      ;DO COMMON ERR MESSAGES.
               ;
               ; PRINT THE MESSAGE AT H&L UNTIL A ZERO.
               ;
CC8C 7E        PMSG:     MOV  A,M       ;GET A CHARACTER.
CC8D B7                  ORA  A         ;IF IT'S ZERO,
CC8E C8                  RZ             ;RETURN.
CC8F 4F                  MOV  C,A       ;OTHERWISE,
CC90 CDE8CA              CALL CONOT     ;PRINT IT.
CC93 23                  INX  H         ;INCREMENT H&L,
CC94 C38CCC              JMP  PMSG      ;AND GET ANOTHER.
               ;
               ; CBIOS MESSAGES
F000 =         RENT     EQU     OF000H  ;MONITOR ENTRY
               ;
               ;
CC97 4E4F542052NRMSG:   DB   'NOT READY ',0
CCA2 5245434F52RNMSG:   DB   'RECORD NOT FOUND ',0
CCB4 4352432000CRCMSG:  DB   'CRC ',0
CCB9 4C4F535420LDMSG:   DB   'LOST DATA ',0
CCC4 4255535920BSYMSG:  DB   'BUSY ',0
CCCA 50524F5445WPMSG:   DB   'PROTECT ',0
CCD3 4641554C54WFMSG:   DB   'FAULT ',0
CCDA 4552524F52ERRMSG:  DB   'ERROR.',0
CCE1 0D0A524541RDMSG:   DB   ODH,OAH,'READ ',0
CCE9 0D0A575249WTMSG:   DB   ODH,OAH,'WRITE ',0
CCF2 0D0A534545SKMSG:   DB   ODH,OAH,'SEEK ',0
CCFA 0D0A484F4DHEMSG:   DB   ODH,OAH,'HOME ',0
CD02 0D0A4D4F55MNTMSG:  DB   ODH,OAH,'MOUNT ',0
```

```
                    ;;;;;;;;;;;;;;;;;;;;;;;;
                    ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                    ;     PRINT 8 BIT WORD IN BINARY FORMAT
                    ;         INPUT:  DATA IN REG A
                    ;
                    ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                    ;
CD0B 47             BITS:   MOV     B,A     ; DATA
CD0C 3E80                   MVI     A,80H   ; MASK
CD0E 0E30           OVER:   MVI     C,30H
CD10 5F                     MOV     E,A     ; STORE MASK
CD11 A0                     ANA     B       ; AND WITH MASK
CD12 CA17CD                 JZ      PRNT    ; JUMP IF ZERO
CD15 0E31                   MVI     C,31H
CD17 CDE8CA         PRNT:   CALL    CONOT
CD1A A0                     ANA     B       ; ZERO CARRY
CD1B 7B                     MOV     A,E     ; LOAD MASK
CD1C 1F                     RAR
CD1D D20ECD                 JNC     OVER
CD20 C9                     RET
                    ;;
                    ;
CD21 0E20           BLNK:   MVI     C,20H             ;PRINT BLANKS, # IN REG. D
CD23 CDEECA         LP1:    CALL    CONOT1
CD26 15                     DCR     D
CD27 C223CD                 JNZ     LP1
CD2A C9                     RET
                    ;
CD2B 7A             BINHA:  MOV     A,D
CD2C 1F                     RAR
CD2D 1F                     RAR
CD2E 1F                     RAR
CD2F 1F                     RAR
CD30 CD48CD                 CALL    BIN1
CD33 4F                     MOV     C,A
CD34 CDE8CA                 CALL    CONOT
CD37 7A                     MOV     A,D
CD38 CD48CD                 CALL    BIN1
CD3B 4F                     MOV     C,A
CD3C CDE8CA                 CALL    CONOT
CD3F C9                     RET
                    ;
                    ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                    ;     OUTPUTS FOUR HEX DIGITS IN ASCII
                    ;     ENTER WITH DATA IN REG PAIR E AND D
                    ;;;;;;;;;;;;;;;;;;;;
                    ;;
                    ;
CD40 CD2BCD         BINB:   CALL    BINHA
CD43 53                     MOV     D,E
CD44 CD2BCD                 CALL    BINHA
CD47 C9                     RET
                    ;
                    ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                    ;     CONVERTS HEX TO ASCII
                    ;     INPUT: 4 BITS HEX  REG A
                    ;     OUTPUT:  8 BIT ASSCII REG A
                    ;
                    ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                    ;;;
                    ;
CD48 E60F           BIN1:   ANI     0FH
CD4A C630                   ADI     30H
CD4C FE3A                   CPI     3AH
CD4E D8                     RC
CD4F C607                   ADI     07H
CD51 C9                     RET
```

```
                    ;
                    ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                    ;      INITIATE SIO PORTS
                    ;
                    ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                    ;;
                    ;
CD52 3EAA           INITA:  MVI     A,0AAH              ;GET DUMMY MODE WORD
CD54 D303                    OUT    CSTAT               ;OUTPUT IT
CD56 3E40                   MVI     A,40H               ;GET RESET BIT
CD58 D303                   OUT     CSTAT               ;RESET SIO BOARD
CD5A 3ECE                   MVI     A,0CEH              ;GET REAL MODE WORD
CD5C D303                   OUT     CSTAT               ;SET THE MODE FOR REAL
CD5E 3E37                   MVI     A,37H               ;GET THE COMMAND
CD60 D303                   OUT     CSTAT               ;OUTPUT IT
CD62 C9                     RET
                    ;
                    ;
                    ;
CD63 0E0D           CRLF:   MVI     C,13                ;CR
CD65 CDE8CA                 CALL    CONOT
CD68 0E0A           LF:     MVI     C,10                ;LF
CD6A CDEECA                 CALL    CONOT1
CD6D 0E7F                   MVI     C,7FH
CD6F CDEECA                 CALL    CONOT1
CD72 CDE8CA                 CALL    CONOT
CD75 C9                     RET
                    ;
                    ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                    ;
                    ;
                    ;
                    ;
                    ;
                    ; DISC TEST FULL TRACK WRITE
                    ;   SELECT TRACK IN SENSE SWITCHES
                    ;    PROM VERSION
                    ;    BRIAN DONLAN
                    ;    JUNE 79
                    ;
CD80                        ORG     0CD80H
                    ;

CD80 C38BCD         ENTRYA: JMP     STARTA
CD83 218000         ENTRYB: LXI     H,080H
CD86 F9                     SPHL                        ;SET STACK
CD87 F3                     DI
CD88 CD52CD                 CALL    INITA               ;RESET SIO
CD8B 21F7CD         STARTA: LXI     H,MSG1B
CD8E CD8CCC         READT:  CALL    PMSG
CD91 3E00                   MVI     A,00H
CD93 320800                 STA     ERRFLG              ;ERROR FLAG OTHER TEST
CD96 CDDACA                 CALL    CONIN               ;READ KEYBOARD
CD99 FE59                   CPI     'Y'
CD9B 2139CE                 LXI     H,MSG2A
CD9E C28ECD                 JNZ     READT
CDA1 CD08CB                 CALL    HOME
CDA4 B7             STARTB: ORA     A                   ;SET FLAGS
CDA5 C28BCD                 JNZ     STARTA              ;ERROR START OVER
CDA8 21A4CD                 LXI     H,STARTB            ;SUBROUTINE RETURN
CDAB E5                     PUSH    H
CDAC DBFF           STARTC: IN      0FFH                ;READ SENSE SWITCHES
CDAE FE4D                   CPI     77                  ;PREVENT TRACK OVER-DRIVE
CDB0 D2C8CD                 JNC     ERRA
CDB3 CD72CC         SEEKA:  CALL    SEEK                ;MOVE HEAD TO TRACK
CDB6 06FF                   MVI     B,0FFH              ;TEST PATTERN
CDB8 3EF4                   MVI     A,0F4H              ;WRITE TRK COMMAND
CDBA D3F8                   OUT     DCOM
CDBC DBFC           WRTLP:  IN      WAIT
CDBE B7                     ORA     A
CDBF F251CC                 JP      WDONE
CDC2 78                     MOV     A,B
CDC3 D3FB                   OUT     DDATA
CDC5 C3BCCD                 JMP     WRTLP
```

```
                  ;
CDC8 06F0         ERRA:   MVI     B,0F0H
CDCA 78           ERRLP:  MOV     A,B
CDCB 2F                   CMA
CDCC D3FF                 OUT     0FFH
CDCE 210000               LXI     H,000H              ;DELAY LOOP
CDD1 110100               LXI     D,01H
CDD4 19           ERRLPB: DAD     D
CDD5 D2D4CD               JNC     ERRLPB
CDD8 47                   MOV     B,A
CDD9 DBFF                 IN      0FFH                ;SEE IF SWITCHES FIXED
CDDB FE4D                 CPI     77
CDDD D2CACD               JNC     ERRLP
CDE0 210000       DELAY:  LXI     H,0H
CDE3 110100               LXI     D,01
CDE6 19           DELP:   DAD     D
CDE7 D2E6CD               JNC     DELP
CDEA 210000       DELAYA: LXI     H,0H
CDED 110100               LXI     D,01H
CDF0 19           DELPA:  DAD     D
CDF1 D2F0CD               JNC     DELPA
CDF4 C3ACCD               JMP     STARTC
                  ;;
CDF7 0D0A0A4449MSG1B:     DB      0DH,0AH,0AH,'DISK TRACK WRITE ROUTINE'
CE12 0D0A4C4F41           DB      0DH,0AH,'LOAD SCRATCH DISK  TYPE Y WHEN READY',0
CE39 0D0A3F3F00MSG2A:     DB      0DH,0AH,'??',0
                  ;
                  ;
                  ;
                  ; DISC TEST FULL TRACK READ
                  ;   SELECT TRACK IN SENSE SWITCHES
                  ;    PROM VERSION
                  ;    BRIAN DONLAN
                  ;    JUNE 79
                  ;
CE40                      ORG     0CE40H
                  ;

CE40 C34BCE       ENTRYD: JMP     STARTD
CE43 218000       ENTRYE: LXI     H,080H
CE46 F9                   SPHL                        ;SET STACK
CE47 F3                   DI
CE48 CD52CD               CALL    INITA               ;RESET SIO
CE4B 21C0CE       STARTD: LXI     H,MSG1E
CE4E CD8CCC       READU:  CALL    PMSG
CE51 3E00                 MVI     A,00H
CE53 320800               STA     ERRFLG
CE56 CDDACA               CALL    CONIN               ;READ KEYBOARD
CE59 FE59                 CPI     'Y'
CE5B 2139CE               LXI     H,MSG2A
CE5E C24ECE               JNZ     READU
CE61 CD08CB               CALL    HOME
CE64 B7           STARTE: ORA     A                   ;SET FLAGS
CE65 C24BCE               JNZ     STARTD              ;ERROR START OVER
CE68 2164CE               LXI     H,STARTE            ;SUBROUTINE RETURN
CE6B E5                   PUSH    H
CE6C DBFF         STARTF: IN      0FFH                ;READ SENSE SWITCHES
CE6E FE4D                 CPI     77                  ;PREVENT TRACK OVER-DRIVE
CE70 D285CE               JNC     ERRD
CE73 CD72CC       SEEKD:  CALL    SEEK                ;MOVE HEAD TO TRACK
CE76 3EE5                 MVI     A,0E5H              ;READ COMMAND
CE78 D3F8                 OUT     DCOM
CE7A DBFC         RDLP :  IN      WAIT
CE7C B7                   ORA     A
CE7D F2B4CE               JP      RDONE
CE80 DBFB                 IN      DDATA
CE82 C37ACE               JMP     RDLP
```

```
                    ;
CE85 06F0           ERRD:   MVI     B,0F0H
CE87 78             ERRLPD: MOV     A,B
CE88 2F                     CMA
CE89 D3FF                   OUT     0FFH
CE8B 110100                 LXI     D,01H
CE8E 210000                 LXI     H,000H          ;DELAY LOOP
CE91 19             ERRLPE: DAD     D
CE92 D291CE                 JNC     ERRLPE
CE95 47                     MOV     B,A
CE96 DBFF                   IN      0FFH            ;SEE IF SWITCHES FIXED
CE98 FE4D                   CPI     77
CE9A D287CE                 JNC     ERRLPD
CE9D 210000         DELAYD: LXI     H,0
CEA0 110100                 LXI     D,01H
CEA3 19             DELPD:  DAD     D
CEA4 D2A3CE                 JNC     DELPD
CEA7 210000         DELAYE: LXI     H,0
CEAA 110100                 LXI     D,01H
CEAD 19             DELPE:  DAD     D
CEAE D2ADCE                 JNC     DELPE
CEB1 C36CCE                 JMP     STARTF
                    ;
CEB4 DBF8           RDONE:  IN      DSTAT
CEB6 E69D                   ANI     9DH
CEB8 57                     MOV     D,A
CEB9 C8                     RZ
CEBA 21E1CC                 LXI     H,RDMSG
CEBD C374CB                 JMP     ERMSG
                    ;;
CEC0 0D0A0A4449MSG1E:       DB      0DH,0AH,0AH,'DISK TRACK READ ROUTINE'
CEDA 0D0A4C4F41             DB      0DH,0AH,'LOAD SCRATCH DISK  TYPE Y WHEN READY ',0
```

```
D000                   ORG     0D000H
               ;       UNIBUS PORT TEST
               ;       AND UBIBUS COMM. TEST COMBINED
               ;         AND SNAP-SHOT
               ;   PROM VERSION
               ;
               ;
               ;
D000 218000    ENTRY1: LXI     H,080H
D003 F9                SPHL            ;SET STACK POINTER
D004 2104D0    ENTRY:  LXI     H,ENTRY
D007 E5                PUSH    H
D008 F3                DI
D009 CDDAD2            CALL    INITA          ;RESET IO
D00C 2184D0            LXI     H,MSG1         ;OPENING MESSAGE
D00F CD50D2            CALL    PMSG
               ;
               ;BEGINING OF TEST
D012 3E01              MVI     A,01H
D014 0E10              MVI     C,10H          ;PORT UNDER TEST
D016 D310      PORT10: OUT     10H
D018 47                MOV     B,A            ;SAVE TEST PATTERN
D019 DB10              IN      10H            ;READ BUSS
D01B B8                CMP     B              ;COMPARE
D01C C45FD0            CNZ     ERR            ;CALL IF IN ERROR
D01F 07                RLC
D020 D216D0            JNC     PORT10         ;TEST FOR A COMPLETE CICLE
               ;
D023 3E01              MVI     A,01H
D025 0E11              MVI     C,11H          ;PORT 11
D027 D311      PORT11: OUT     11H
D029 47                MOV     B,A            ;SAVE PATTERN
D02A DB11              IN      11H            ;READ BUSS
D02C B8                CMP     B              ;C OMPARE
D02D C45FD0            CNZ     ERR            ;CALL IF ERROR
D030 07                RLC
D031 D227D0            JNC     PORT11
               ;
               ;
D034 3E01              MVI     A,01H
D036 0E12              MVI     C,12H          ;PORT12
D038 D312      PORT12: OUT     12H
D03A 47                MOV     B,A            ;SAVE TEST PATERN
D03B DB12              IN      12H
D03D B8                CMP     B
D03E C45FD0            CNZ     ERR
D041 07                RLC
D042 D238D0            JNC     PORT12
               ;
               ;
D045 3E01              MVI     A,01H
D047 0E13              MVI     C,13H
D049 D313      PORT13: OUT     13H
D04B 47                MOV     B,A
D04C DB13              IN      13H
D04E B8                CMP     B
D04F C45FD0            CNZ     ERR
D052 07                RLC
D053 D249D0            JNC     PORT13
               ;
               ;
               ;
               ;
D056 21D5D0            LXI     H,MSG4         ;FINISHED MESSAGE
D059 CD50D2            CALL    PMSG
D05C C300F0            JMP     RENT           ;RETURN TO MONITOR
```

```
                   ;
D05F C5            ERR:     PUSH    B                   ;SAVE ERROR PATTERN
D060 F5                     PUSH    A                   ;SAVE TEST PATTERN
D061 51                     MOV     D,C
D062 2198D0                 LXI     H,MSG2
D065 CD50D2                 CALL    PMSG
D068 CD7BD2                 CALL    BINHA               ;PRINT 2 DIGITS
D06B 21AED0                 LXI     H,MSG0              ;ERROR MESSAGE
D06E CD50D2                 CALL    PMSG
D071 78                     MOV     A,B                 ;LOAD TEST PATTERN
D072 CD5BD2                 CALL    BITS                ;PRINT TEST PATTERN
D075 21C1D0                 LXI     H,MSG3              ;MORE TEXT
D078 CD50D2                 CALL    PMSG
D07B F1                     POP     A
D07C CD5BD2                 CALL    BITS                ;PRINT ERROR PATTERN
D07F C1                     POP     B                   ;RESTORE B AND C
D080 78                     MOV     A,B                 ;MOVE TEST PATTERN TO A
D081 37                     STC
D082 3F                     CMC
D083 C9                     RET
                   ;
                   ;
D084 0A0A0D554EMSG1         DB      0AH,0AH,0DH,'UNIBUS PORT TEST',0
D098 0A0A0D4552MSG2:        DB      0AH,0AH,0DH,'ERROR PORT NO.     ',0
D0AE 0A0D544553MSG0:        DB      0AH,0DH,'TEST PATTERN    ',0
D0C1 0A0D414354MSG3         DB      0AH,0DH,'ACTUAL PATTERN   ',0
D0D5 0A0A0D454EMSG4         DB      0AH,0AH,0DH,'END OF TEST       ',0
F000 =             RENT     EQU     0F000H              ;MONITOR ENTRY
                   ;
                   ;
                   ;UNIBUS COMMUNICATION TEST
                   ;
                   ;
D100                        ORG     0D100H
D100 218000        ENTRY2: LXI     H,080H
D103 F9                     SPHL            ;SET STACK
D104 CDDAD2                 CALL    INITA
D107 2107D1        ENTRY3: LXI     H,ENTRY3
D10A E5                     PUSH    H
D10B F3                     DI
D10C 2160D1                 LXI     H,MSG5              ;OPENNING MEESSAGE
D10F CD50D2                 CALL    PMSG
D112 CDA2D2                 CALL    BBIN                ;GET HEX CHAR
D115 D5                     PUSH    D                   ;SAVE ADDRESS
D116 2196D1                 LXI     H,MSG6              ;REQUEST MODE
D119 CD50D2                 CALL    PMSG
D11C CD22D2        TRYGN:   CALL    CONIN
D11F FE49                   CPI     'I'
D121 CA3ED1                 JZ      PUTIN               ;JUMP IF INPUT MAODE
D124 FE03                   CPI     03H                 ;TEST IF CONTROL C
D126 CA00F0                 JZ      RENT                ;RETURN TO MONITOR
D129 FE4F                   CPI     'O'
D12B C257D1                 JNZ     QUEST
                   ;
                   ;OUTPUT MODE
                   ;
D12E 21C7D1        PUTOUT: LXI     H,MSG11             ;OUTPUT MESSAGE
D131 CD50D2                 CALL    PMSG
D134 CDA2D2                 CALL    BBIN                ;GET DIGITS TO OUTPUT
D137 C1                     POP     B                   ;RESTORE ADDRESS TO REG B & C
D138 CD08D3                 CALL    DATAO               ;UNIBUSS DRIVER
D13B C34ED1                 JMP     DONE
                   ;
D13E 21F1D1        PUTIN:   LXI     H,MSG9              ;INPUT MESAGE
D141 CD50D2                 CALL    PMSG
D144 C1                     POP     B                   ;RESTORE ADDRESS TO B & C
D145 CDFED2                 CALL    DATAI               ;UNIBUS INPUT ROUTINE
D148 CD90D2                 CALL    BINB                ;PRINT DATA FROM BUSS
D14B C34ED1                 JMP     DONE
                   ;
                   ;
D14E 2105D2        DONE:    LXI     H,MSG10
D151 CD50D2                 CALL    PMSG                ;PRINT END OF TEST
D154 C307D1                 JMP     ENTRY3
```

```
               ;
D157 21C2D1    QUEST:, LXI     H,MSG7
D15A CD50D2            CALL    PMSG             ; ??
D15D C31CD1            JMP     TRYGN
               ;
               ;
D160 0A0A0D554EMSG5:   DB      0AH,0AH,0DH,'UNIBUS COMMUNICATION TEST'
D17C 0A0D454E54        DB      0AH,0DH,'ENTER UNIBUS ADDRESS   ',0
D196 0A0D494E50MSG6:   DB      0AH,0DH,'INPUT (I), OUTPUT (O), EXIT (CONTROL C) ?',0
D1C2 0A0D203F00MSG7:   DB      0AH,0DH,' ?',0
D1C7 0A0D454E54MSG11:  DB      0AH,0DH,'ENTER DATA TO OUTPUT IN 4 HEX DIGITS   ',0
D1F1 0A0D204441MSG9:   DB      0AH,0DH,' DATA FROM BUS   ',0
D205 0A0D545241MSG10   DB      0AH,0DH,'TRANSFER COMPLETE',0
               ;
               ;

               ; diagnostic input output routines
               ; for brian donlan  26 feb 79

0003 =         CSTAT   EQU  3          ;CONSOLE STATUS PORT.
0003 =         CCOM    EQU  3          ;CONSOLE COMMAND PORT.
0002 =         CDATA   EQU  2          ;CONSOLE DATA PORT.
0002 =         CKBR    EQU  00000010B  ;KEYBOARD READY BIT.
0001 =         CPTR    EQU  00000001B  ;PRINT READY BIT.
0001 =         CNULL   EQU  1          ;CONSOLE NULL COUNT.

               ; CHECK CONSOLE INPUT STATUS.
               ;

D219 DB03      CONST:  IN   CSTAT      ;READ CONSOLE STATUS.
D21B E602              ANI  CKBR       ;LOOK AT KB READY BIT.
D21D 3E00              MVI  A,0        ;SET A=0 FOR RETURN.
D21F C8                RZ              ;NOT READY WHEN ZERO.
D220 2F                CMA             ;IF READY A=FF.
D221 C9                RET             ;RETURN FROM CONST.

               ;
               ; READ A CHARACTER FROM CONSOLE.
               ;
D222 DB03      CONIN:   IN   CSTAT              ;READ CONSOLE STATUS.
D224 E602               ANI  CKBR      ;IF NOT READY,

D226 CA22D2            JZ   CONIN      ;READY WHEN HIGH.
D229 DB02               IN   CDATA     ;READ A CHARACTER.
D22B D302              OUT     CDATA
D22D E67F               ANI  7FH       ;MAKE MOST SIG. BIT = 0.
D22F C9                 RET
               ;
               ; WRITE A CHARACTER TO THE CONSOLE DEVICE.
               ;
D230 3E0D      CONOT:  MVI  A,0DH      ;IF IT'S A CR,
D232 B9                CMP  C          ;THEN HOP OUT
D233 CA41D2            JZ   CONUL      ;TO NULL ROUTINE.
D236 DB03      CONOT1: IN   CSTAT      ;READ CONSOLE STATUS.
D238 E601              ANI  CPTR       ;IF NOT READY,
D23A CA36D2            JZ   CONOT1     ;READY WHEN HIGH.
D23D 79                MOV  A,C        ;GET CHARACTER.
D23E D302              OUT  CDATA      ;PRINT IT.
D240 C9                RET             ;RETURN.
D241 C5        CONUL:  PUSH B          ;SAVE B&C.
D242 0601              MVI  B,CNULL    ;GET NULL COUNT.
D244 CD36D2    CONUL1: CALL CONOT1     ;PRINT CR.
D247 0E00              MVI  C,0        ;GET NULL CHAR.
D249 05                DCR  B          ;DECREMENT COUNTER.
D24A C244D2            JNZ  CONUL1     ;DO NEXT NULL.
D24D C1                POP  B          ;RESTORE B&C.
D24E 79                MOV  A,C        ;RESTORE A.
D24F C9                RET             ;RETURN.
```

```
               ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
               ;       PRINT MESSAGE UNTIL ZERO
               ;       MESSAGE ADDRESS REG H & L
               ;
               ;;;;;;;;;;;;;;;;;;;;;
D250 7E        PMSG:   MOV     A,M     ;GET CHAR
D251 B7                ORA     A       ;IS IT A  ZERO
D252 C8                RZ
D253 4F                MOV     C,A     ;OTHERWISE PRINT
D254 CD30D2            CALL    CONOT
D257 23                INX     H       ;INC ADDRESSS
D258 C350D2            JMP     PMSG
               ;
               ;
               ;
               ;;;;;;;;;;;;;;;;;;;;;;
               ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
               ;    PRINT 8 BIT WORD IN BINARY FORMAT
               ;       INPUT:  DATA IN REG A
               ;
               ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
               ;
D25B 47        BITS:   MOV     B,A     ; DATA
D25C 3E80              MVI     A,80H   ; MASK
D25E 0E30      OVER:   MVI     C,30H
D260 5F                MOV     E,A     ; STORE MASK
D261 A0                ANA     B       ; AND WITH MASK
D262 CA67D2            JZ      PRNT    ; JUMP IF ZERO
D265 0E31              MVI     C,31H
D267 CD30D2    PRNT:   CALL    CONOT
D26A A0                ANA     B       ; ZERO CARRY
D26B 7B                MOV     A,E     ; LOAD MASK
D26C 1F                RAR
D26D D25ED2            JNC     OVER
D270 C9                RET
               ;;
               ;
D271 0E20      BLNK:   MVI     C,20H           ;PRINT BLANKS, # IN REG. D
D273 CD36D2    LP17:   CALL    CONOT1
D276 15                DCR     D
D277 C273D2            JNZ     LP17
D27A C9                RET
               ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
               ;
               ;     OUTPUTS 2 HEX DIGITS IN ASCCII
               ;            FROM REG D
               ;
               ;;;;;;;;;;;;
               ;
               ;
D27B 7A        BINHA:  MOV     A,D
D27C 1F                RAR
D27D 1F                RAR
D27E 1F                RAR
D27F 1F                RAR
D280 CD98D2            CALL    BIN1
D283 4F                MOV     C,A
D284 CD30D2            CALL    CONOT
D287 7A                MOV     A,D
D288 CD98D2            CALL    BIN1
D28B 4F                MOV     C,A
D28C CD30D2            CALL    CONOT
D28F C9                RET
               ;
               ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
               ;    OUTPUTS FOUR HEX DIGITS IN ASCII
               ;    ENTER WITH DATA IN REG PAIR E AND D
               ;;;;;;;;;;;;;;;;;;
               ;;
               ;
D290 CD7BD2    BINB:   CALL    BINHA
D293 53                MOV     D,E
D294 CD7BD2            CALL    BINHA
D297 C9                RET
```

```
                ;
                ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                ;    CONVERTS HEX TO ASCII
                ;    INPUT: 4 BITS HEX  REG A
                ;    OUTPUT:  8 BIT ASSCII REG A
                ;
                ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                ;;;
                ;
D298 E60F       BIN1:   ANI     0FH
D29A C630               ADI     30H
D29C FE3A               CPI     3AH
D29E D8                 RC
D29F C607               ADI     07H
D2A1 C9                 RET
                ;
                ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                ;    INPUTS 4 DIGITS FROM CONSOLE
                ;    RETURN;  4 HEX DIGITS IN REG E-D
                ;
                ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                ;;
                ;
D2A2 CD22D2     BBIN:   CALL    CONIN
D2A5 CDD1D2             CALL    AHS1
D2A8 17                 RAL
D2A9 17                 RAL
D2AA 17                 RAL
D2AB 17                 RAL
D2AC E6F0               ANI     0F0H
D2AE 57                 MOV     D,A
D2AF CD22D2             CALL    CONIN
D2B2 CDD1D2             CALL    AHS1
D2B5 E60F               ANI     0FH
D2B7 B2                 ORA     D
D2B8 57                 MOV     D,A
D2B9 CD22D2             CALL    CONIN
D2BC CDD1D2             CALL    AHS1
D2BF 17                 RAL
D2C0 17                 RAL
D2C1 17                 RAL
D2C2 17                 RAL
D2C3 E6F0               ANI     0F0H
D2C5 5F                 MOV     E,A
D2C6 CD22D2             CALL    CONIN
D2C9 CDD1D2             CALL    AHS1
D2CC E60F               ANI     0FH
D2CE B3                 ORA     E
D2CF 5F                 MOV     E,A
D2D0 C9                 RET
                ;
                ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                ;    CONVERT ASCII TO HEX
                ;    INPUT;  8 BIT ASCII  REG A
                ;    OUTPUT:  4 BIT HEX REG A
                ;
                ;;;;;;;;;;;;;;;;;;;;;;;;;;;
                ;;
                ;
D2D1 00         AHS1:   NOP
D2D2 D630               SUI     30H
D2D4 FE0A               CPI     0AH
D2D6 D8                 RC
D2D7 D607               SUI     07H
D2D9 C9                 RET
                ;
                ;;;;;;;;;;
                ;
```

```
               ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
               ;     INITIATE SIO PORTS
               ;
               ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
               ;;
               ;
D2DA 3EAA      INITA:  MVI     A,0AAH           ;GET DUMMY MODE WORD
D2DC D303               OUT    CSTAT            ;OUTPUT IT
D2DE 3E40              MVI     A,40H            ;GET RESET BIT
D2E0 D303              OUT     CSTAT            ;RESET SIO BOARD
D2E2 3ECE              MVI     A,0CEH           ;GET REAL MODE WORD
D2E4 D303              OUT     CSTAT            ;SET THE MODE FOR REAL
D2E6 3E37              MVI     A,37H            ;GET THE COMMAND
D2E8 D303              OUT     CSTAT            ;OUTPUT IT
D2EA C9                RET
               ;
               ;
               ;
D2EB 0E0D      CRLF:   MVI     C,13             ;CR
D2ED CD30D2            CALL    CONOT
D2F0 0E0A      LF:     MVI     C,10             ;LF
D2F2 CD36D2            CALL    CONOT1
D2F5 0E7F              MVI     C,7FH
D2F7 CD36D2            CALL    CONOT1
D2FA CD30D2            CALL    CONOT
D2FD C9                RET
               ;
               ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
               ;
               ;
D2FE CD91D3    DATAI:  CALL    GETBUS
D301 CD12D3            CALL    DATI
D304 97                SUB     A
D305 D315              OUT     15H
D307 C9                RET
               ;
               ;
D308 CD91D3    DATAO:  CALL    GETBUS
D30B CD51D3            CALL    DATO
D30E 97                SUB     A
D30F D315              OUT     15H
D311 C9                RET
               ;
               ;
               ;
               ;ROUTINE TO INPUT A 16 BIT WORD FROM UNIBUS
               ;REG B = A<16:09>, REG C = A<08:01>
               ;DATA WILL BE CONTAINED IN REG D = D<15:08>, REG C = D<07:00>
               ;
               ;
D312 3EFF      DATI:   MVI     A,0FFH           ;SET LOOP COUNT
D314 328200            STA     BIZCNT
D317 DB14      BIZLP1: IN      14H              ;CHECK FOR SYS = 0
D319 E604              ANI     04H              ;FROM LAST TRANSACTION
D31B C2A2D3            JNZ     BBUSY1
               ;
D31E 78                MOV     A,B              ;OUTPUT HIGH ADDRESS
D31F D310              OUT     10H
D321 79                MOV     A,C              ;OUTPUT LOW ADDRESS
D322 D311              OUT     11H
D324 97                SUB     A                ;OUTPUT C1=0
D325 D314              OUT     14H
D327 F601              ORI     01H              ;OUTPUT MSYN=1
D329 D314              OUT     14H
               ;
D32B 3EFF      SYNLP1: MVI     A,0FFH           ;LOOP COUNT
D32D 328100            STA     SYNCNT
D330 DB14      DILOOP: IN      14H              ;CHECKS IF SSYN=1
D332 D3FF              OUT     0FFH
D334 E604              ANI     04H
D336 CAC8D3            JZ      NOSYN1
```

```
                       ;
D339 DB12                        IN      12H              ;INPUT HIGH DATA
D33B 57                          MOV     D,A
D33C DB13                        IN      13H              ;INPUT LOW DATA
D33E 5F                          MOV     E,A
                       ;
D33F 97                          SUB     A
D340 D314                        OUT     14H              ;CLEARS MSYN
D342 D310                        OUT     10H              ;AND EVERYTHING
D344 D311                        OUT     11H              ;PUT OUT TO BUS
D346 D312                        OUT     12H
D348 D313                        OUT     13H
D34A DBFF                        IN      0FFH
D34C B7                          ORA     A                ;SET FLAGS
D34D C212D3                      JNZ     DATI             ;LOOP IF SENSE SWITCH UP
D350 C9                          RET
                       ;
                       ;
                       ;ROUTINE TO OUTPUT A 16 BIT WORD ON THE UNIBUS
                       ;REG B = A<16:09>, REG C = A<08:01>
                       ;REG D = D<15:08>, REG E = D<07:00>
                       ;
D351 3EFF              DATO:     MVI     A,0FFH
D353 328200                      STA     BIZCNT
D356 DB14              BIZLP2:   IN      14H
D358 E604                        ANI     04H
D35A C2B5D3                      JNZ     BBUSY2
                       ;
D35D 78                          MOV     A,B              ;OUTPUT HIGH ADDRESS
D35E D310                        OUT     10H
D360 79                          MOV     A,C              ;OUTPUT LOW ADDRESS
D361 D311                        OUT     11H
D363 7A                          MOV     A,D              ;OUTPUT HIGH DATA
D364 D312                        OUT     12H
D366 7B                          MOV     A,E              ;OUTPUT LOW DATA
D367 D313                        OUT     13H
D369 3E02                        MVI     A,02H            ;OUTPUT C1=1
D36B D314                        OUT     14H
                       ;
D36D 3E03                        MVI     A,03H            ;OUTPUT MSYN=1
D36F D314                        OUT     14H
                       ;
D371 3EFF              SYNLP2:   MVI     A,0FFH
D373 328100                      STA     SYNCNT
D376 DB14              DOLOOP:   IN      14H              ;CHECKS FOR SSYN
D378 D3FF                        OUT     0FFH
D37A E604                        ANI     04H              ;TO GET ASSERTED
D37C CAF8D3                      JZ      NOSYN2
                       ;
D37F 97                          SUB     A
D380 D314                        OUT     14H              ;CLEARS MSYN AND C1
D382 D310                        OUT     10H
D384 D311                        OUT     11H              ;CLEARS EVERYTHING
D386 D312                        OUT     12H              ;OUTPUT TO THE BUS
D388 D313                        OUT     13H
D38A DBFF                        IN      0FFH             ;READ SENSE SWITCH
D38C B7                          ORA     A                ;SET FLAGS
D38D C251D3                      JNZ     DATO             ;LOOP IF UP
D390 C9                          RET
                       ;
                       ;
                       ;
                       ;
D391 3EFF              GETBUS:   MVI     A,0FFH
D393 328000                      STA     GETCNT
D396 3E01                        MVI     A,01H
D398 D315                        OUT     15H
D39A DB15              LOOP:     IN      15H
D39C E601                        ANI     01H
D39E CA0CD4                      JZ      NOGET
D3A1 C9                          RET
```

```
                        ;  ON-LINE UNIBUS DIAGNOSTICS
                        ;    BY BRIAN DONLAN  24 APR 79
                        ;
D3A2 3A8200     BBUSY1: LDA BIZCNT                  ;LOOP COUNT
D3A5 3D                 DCR     A
D3A6 328200             STA     BIZCNT              ;NEW COUNT
D3A9 C217D3             JNZ     BIZLP1              ;JUMP IF STILL COUNT
D3AC 213ED4             LXI     H,ERMSG2
D3AF CD50D2             CALL    PMSG    ;DISPLAY ERROR MESSAGE
D3B2 C307D1             JMP     ENTRY3
                ;
                ;
D3B5 3A8200     BBUSY2: LDA     BIZCNT
D3B8 3D                 DCR     A
D3B9 328200             STA     BIZCNT
D3BC C256D3             JNZ     BIZLP2
D3BF 213ED4             LXI     H,ERMSG2
D3C2 CD50D2             CALL    PMSG
D3C5 C307D1             JMP     ENTRY3
                ;
                ;
                ;
                ;
D3C8 3A8100     NOSYN1: LDA     SYNCNT
D3CB 3D                 DCR     A
D3CC 328100             STA     SYNCNT
D3CF C230D3             JNZ     DILOOP
D3D2 DBFF               IN      OFFH
D3D4 E640               ANI     040H
D3D6 C230D3             JNZ     DILOOP
D3D9 50         SYSERR: MOV     D,B                 ;
D3DA 59                 MOV     E,C                 ;MOV ADDRESS FOR OUTPUT
D3DB 215BD4             LXI     H,ERMSG3
D3DE CD50D2             CALL    PMSG
D3E1 CD90D2             CALL    BINB                ;OUTPUT ADDRESS
D3E4 216BD4             LXI     H,ERMSG4
D3E7 CD50D2             CALL    PMSG
D3EA AF                 XRA     A                   ;ZERO A
D3EB D311               OUT     11H
D3ED D312               OUT     12H
D3EF D313               OUT     13H
D3F1 D314               OUT     14H
D3F3 D310               OUT     10H
D3F5 C307D1             JMP     ENTRY3
                ;
D3F8 3A8100     NOSYN2: LDA     SYNCNT
D3FB 3D                 DCR     A
D3FC 328100             STA     SYNCNT
D3FF C276D3             JNZ     DOLOOP
D402 DBFF               IN      OFFH
D404 E640               ANI     040H
D406 C276D3             JNZ     DOLOOP
D409 C3D9D3             JMP     SYSERR
                ;
                ;
                ;
D40C 3A8000     NOGET:  LDA     GETCNT
D40F 3D                 DCR     A
D410 328000             STA     GETCNT
D413 C29AD3             JNZ     LOOP
D416 211FD4             LXI     H,ERMSG1
D419 CD50D2             CALL    PMSG
D41C C307D1             JMP     ENTRY3
                ;
                ;
                ;
                ;
                ;
                ;
0080 =          GETCNT: EQU     080H
0081 =          SYNCNT: EQU     081H
0082 =          BIZCNT: EQU     082H
D41F 0D0D0A2020ERMSG1:  DB      ODH,ODH,OAH,'  IMSAI CAN NOT GET BUSS   ',0
D43E 0D0D0A2020ERMSG2:  DB      ODH,ODH,OAH,'  ERROR BUSS BUSY          ',0
D45B 0D0A202044ERMSG3:  DB      ODH,OAH,'  DEVICE NO. ',0
D46B 204E4F2052ERMSG4:  DB      ' NO RESPONSE ',0
```

```
D500               ; UNIBUS SNAP SHOT ROUTINE
                            ORG     0D500H
D500 2100F0        ENTR" :  LXI     H,0F000H
D503 E5                     PUSH    H
D504 F3                     DI
D505 CDDAD2                 CALL    INITA                ;RESET I/O
                   ;
                   ;SUBROUTINE ENTRY POINT
D508 216BD5        ENTRY5:  LXI     H,MSG12
D50B CD50D2                 CALL    PMSG
D50E DB10                   IN      10H                  ;HIGH ADDRESS
D510 57                     MOV     D,A                  ;SAVE IN D
D511 DB11                   IN      11H                  ;LOW ADDRESS
D513 5F                     MOV     E,A
D514 CD90D2                 CALL    BINB                 ;PRINT UNIBUS ADDRESS
                   ;
D517 1608                   MVI     D,08H                ;SPACE OVER
D519 CD71D2                 CALL    BLNK
D51C DB12                   IN      12H                  ;HIGH DATA
D51E 57                     MOV     D,A
D51F DB13                   IN      13H                  ;LOW DATA BITS
D521 5F                     MOV     E,A
D522 CD90D2                 CALL    BINB                 ;PRINT UNIBUS DATA BITS
D525 1608                   MVI     D,08H                ;SPACE OVER
D527 CD71D2                 CALL    BLNK
                   ;
D52A DB14                   IN      14H                  ;STATUS PORT
D52C E604                   ANI     04H                  ;FIND SLAVE SYN
D52E CA36D5                 JZ      NOSIS
D531 0E31                   MVI     C,'1'
D533 C338D5                 JMP     OUTSIS
D536 0E30          NOSIS:   MVI     C,'0'
D538 CD30D2        OUTSIS:  CALL    CONOT                ;PRINT SLAVE SYN
D53B 1609                   MVI     D,09H                ;SPACE OVER
D53D CD71D2                 CALL    BLNK
                   ;
D540 DB15                   IN      15H                  ;STATUS PORT
D542 E601                   ANI     01H                  ;BUS GRANT
D544 CA4CD5                 JZ      NOBUS
D547 0E31                   MVI     C,'1'
D549 C34ED5                 JMP     OUTBUS
D54C 0E30          NOBUS:   MVI     C,'0'
D54E CD30D2        OUTBUS:  CALL    CONOT                ;PRINT BUS GRANT
D551 160B                   MVI     D,0BH                ;SPACE OVER
D553 CD71D2                 CALL    BLNK
                   ;
D556 DB14                   IN      14H
D558 E600                   ANI     00H
D55A CA62D5                 JZ      NOMSYN
D55D 0E31                   MVI     C,'1'
D55F C364D5                 JMP     OUTMSN
D562 0E30          NOMSYN:  MVI     C,'0'
D564 CD30D2        OUTMSN:  CALL    CONOT                ;PRINT MSYN
D567 C9                     RET
D568 C368D5        FINIS:   JMP     FINIS
                   ;
D56B 0A0A0D554EMSG12:       DB      0AH,0AH,0DH,'UNIBUS SNAP-SHOT   '
D580 0A0D414444             DB      0AH,0DH,'ADDRESS          DATA        SSYN        GRANT        MSYN'
D5B4 0A0D202000             DB      0AH,0DH,'  ',0
```

```
                    ;8K MINI MEMORY TEST
                    ;
                    ;
                    ; BRIAN DONLAN
                    ; PROM VERSION
D600                          ORG     0D600H
D600 F3             ENTER:    DI
D601 3EFE                     MVI     A,0FEH
D603 D3FF                     OUT     0FFH            ;OUTPUT PHASE I LITES
D605 210000                   LXI     H,000H          ;START ADDRESS
D608 AF             LP2:      XRA     A               ;ZERO ACC
D609 77             LP1:      MOV     M,A             ;STORE TEST PATTERN IN MEM.
D60A 46                       MOV     B,M             ;READ BACK TO B
D60B B8                       CMP     B               ;COMPARE FOR OK
D60C C26DD6                   JNZ     ERR1            ;JUMP IF ERROR
D60F 3C                       INR     A               ;NEW TEST PATTERN
D610 C209D6                   JNZ     LP1
D613 23                       INX     H
D614 1100E0                   LXI     D,0E000H                ;STOP ADDRESS
D617 EB                       XCHG
D618 19                       DAD     D               ;ADD TWO'S COMPLIMENT
D619 EB                       XCHG
D61A D208D6                   JNC     LP2
                    ;
                    ; PHASE II
D61D 3EFD                     MVI     A,0FDH          ;PHASE II LITES
D61F D3FF                     OUT     0FFH
D621 210000                   LXI     H,000H
D624 74             LP3:      MOV     M,H             ;LOW ADDRESS TO MEM
D625 23                       INX     H
D626 1100E0                   LXI     D,0E000H        ;STOP ADDRESS
D629 EB                       XCHG
D62A 19                       DAD     D
D62B EB                       XCHG
D62C D224D6                   JNC     LP3
                    ;         READ MEMORY
D62F 210000                   LXI     H,000H
D632 7E             LP4:      MOV     A,M             ;READ MEMORY
D633 94                       SUB     H               ;COMPARE
D634 C293D6                   JNZ     ERR2            ;JUMP IF ERROR
D637 23                       INX     H
D638 1100E0                   LXI     D,0E000H
D63B EB                       XCHG
D63C 19                       DAD     D
D63D EB                       XCHG
D63E D232D6                   JNC     LP4
                    ;
                    ; PHASE III
                    ;
D641 3EFC                     MVI     A,0FCH
D643 D3FF                     OUT     0FFH            ;PHASE THREE LITES
D645 210000                   LXI     H,000H
D648 75             LP5:      MOV     M,L             ;STORE HIGH ADDRESS IN ALL  MEM
D649 23                       INX     H
D64A 1100E0                   LXI     D,0E000H
D64D EB                       XCHG
D64E 19                       DAD     D
D64F EB                       XCHG
D650 D248D6                   JNC     LP5
                    ;READ MEM
D653 210000                   LXI     H,000H
D656 7E             LP6:      MOV     A,M             ;READ MEMORY
D657 95                       SUB     L               ;COMPARE
D658 C29FD6                   JNZ     ERR3
D65B 23                       INX     H
D65C 1100E0                   LXI     D,0E000H
D65F EB                       XCHG
D660 19                       DAD     D
D661 EB                       XCHG
D662 D256D6                   JNC     LP6
```

```
                       ;   ALL PHASE COMPLETE
D665 3EFF                      MVI     A,0FFH
D667 2100D6                    LXI     H,ENTER
D66A C3ABD6                    JMP     LITES              ;GO TO LITES PROG
                       ;
                       ;
                       ;PHASE I ERROR
D66D EB                ERR1:   XCHG
D66E 4F                        MOV     C,A                ;SAVE BAD DATA
D66F 2177D6                    LXI     H,COMERR                ;RETURN
D672 3EF1                      MVI     A,0F1H
D674 C3ABD6                    JMP     LITES              ;PHASE I ERROR LITES
                       ;
                       ;
                       ;COMMON ERROR OUTPUT ROUTINE
D677 7A                COMERR: MOV     A,D
D678 217ED6                    LXI     H,LOADD            ;HIGH ADDRESS
D67B C3ABD6                    JMP     LITES              ;RETURN
D67E 7B                LOADD:  MOV     A,E
D67F 2185D6                    LXI     H,TPAT             ;LOW ADDRES TO LITES
D682 C3ABD6                    JMP     LITES              ;RETURN
D685 79                TPAT:   MOV     A,C
D686 218CD6                    LXI     H,ACTDAT           ;TEST PATTERN TO LITES
D689 C3ABD6                    JMP     LITES              ;RETURN
D68C 78                ACTDAT: MOV     A,B
D68D 2100D6                    LXI     H,ENTER            ;ACTUAL DATA TO LITES
D690 C3ABD6                    JMP     LITES                   ;START OVER
                       ;;
                       ;
                       ; PHASE II ERROR
D693 EB                ERR2:   XCHG
D694 82                        ADD     D                  ;SAVE BAD ADDRESS
D695 47                        MOV     B,A
D696 4A                        MOV     C,D
D697 3EF2                      MVI     A,0F2H
D699 2177D6                    LXI     H,COMERR           ;PHASE II ERROR TO LITES
D69C C3ABD6                    JMP     LITES                   ;RETURN
                       ;
                       ;
                       ; PHASE III ERROR
D69F EB                ERR3:   XCHG
D6A0 83                        ADD     E          ;SAVE BAD ADDRESS
D6A1 47                        MOV     B,A
D6A2 4B                        MOV     C,E
D6A3 3EF3                      MVI     A,0F3H
D6A5 2177D6                    LXI     H,COMERR           ;PHASE II ERRO TO LITES
D6A8 C3ABD6                    JMP     LITES              ;RETURN
                       ;
                       ;
                       ;LITES ROUTINE    ENTER WITH RETURN IN REG H&L
                       ;                            DATA FOR LITES IN A
D6AB 2F                LITES:  CMA
D6AC D3FF                      OUT     0FFH               ;OUTPUT LITES
D6AE F9                        SPHL                       ;SAVE RETURN IN SP
D6AF DBFF                      IN      0FFH               ;READ SENSE SWITCHES
D6B1 67                        MOV     H,A                ;SAVE IN H
D6B2 DBFF              LP7:    IN      0FFH               ;READ SWITCHES
D6B4 AC                        XRA     H                  ;SEE IF THEY CHANGED
D6B5 CAB2D6                    JZ      LP7
D6B8 2118FC                    LXI     H,0FC18H
D6BB 23                LP8:    INX     H                        ;DELAY LOOP
D6BC AF                        XRA     A
D6BD B4                        ORA     H
D6BE C2BBD6                    JNZ     LP8
D6C1 210000                    LXI     H,0
D6C4 39                        DAD     SP                 ;ZERO H
D6C5 E9                        PCHL                       ;MOVE RETURN BACK TO H & L
                                                          ;RETURN
```

```
               ; 24K MINI-MEMORY TEST
               ;  PROM VERSION
               ;
               ;
               ;
               ; BRIAN DCNLAN
D700                   ORG     0D700H
D700 F3        ENTER2: DI
D701 3EFE              MVI     A,0FEH
D703 D3FF              OUT     0FFH             ;OUTPUT PHASE I LITES
D705 210000            LXI     H,000H           ;START ADDRESS
D708 AF        LP22:   XRA     A                ;ZERO ACC
D709 77        LP12:   MOV     M,A              ;STORE TEST PATTERN IN MEM.
D70A 46                MOV     B,M              ;READ BACK TO B
D70B B8                CMP     B                ;COMPARE FOR OK
D70C C26DD7            JNZ     ERR12            ;JUMP IF ERROR
D70F 3C                INR     A                ;NEW TEST PATTERN
D710 C209D7            JNZ     LP12
D713 23                INX     H
D714 1100A0            LXI     D,0A000H                 ;STOP ADDRESS
D717 EB                XCHG
D718 19                DAD     D                ;ADD TWO'S COMPLIMENT
D719 EB                XCHG
D71A D208D7            JNC     LP22
               ;
               ; PHASE II
D71D 3EFD              MVI     A,0FDH           ;PHASE II LITES
D71F D3FF              OUT     0FFH
D721 210000            LXI     H,000H
D724 74        LP32:   MOV     M,H              ;LOW ADDRESS TO MEM
D725 23                INX     H
D726 1100A0            LXI     D,0A000H         ;STOP ADDRESS
D729 EB                XCHG
D72A 19                DAD     D
D72B EB                XCHG
D72C D224D7            JNC     LP32
               ;       READ MEMORY
D72F 210000            LXI     H,000H
D732 7E        LP42:   MOV     A,M              ;READ MEMORY
D733 94                SUB     H                ;COMPARE
D734 C293D7            JNZ     ERR22            ;JUMP IF ERROR
D737 23                INX     H
D738 1100A0            LXI     D,0A000H
D73B EB                XCHG
D73C 19                DAD     D
D73D EB                XCHG
D73E D232D7            JNC     LP42
               ;
               ; PHASE III
               ;
D741 3EFC              MVI     A,0FCH
D743 D3FF              OUT     0FFH             ;PHASE THREE LITES
D745 210000            LXI     H,000H
D748 75        LP52:   MOV     M,L              ;STORE HIGH ADDRESS IN ALL  MEM
D749 23                INX     H
D74A 1100A0            LXI     D,0A000H
D74D EB                XCHG
D74E 19                DAD     D
D74F EB                XCHG
D750 D248D7            JNC     LP52
               ;READ MEM
D753 210000            LXI     H,000H
D756 7E        LP62:   MOV     A,M              ;READ MEMORY
D757 95                SUB     L                ;COMPARE
D758 C29FD7            JNZ     ERR32
D75B 23                INX     H
D75C 1100A0            LXI     D,0A000H
D75F EB                XCHG
D760 19                DAD     D
D761 EB                XCHG
D762 D256D7            JNC     LP62
```

```
                  ;
                  ;    ALL PHASE COMPLETE
D765 3EFF                    MVI     A,OFFH
D767 2100D7                  LXI     H,ENTER2
D76A C3ABD7                  JMP     LITES2            ;GO TO LITES PROG
                  ;
                  ;
                  ;PHASE I ERROR
D76D EB           ERR12:     XCHG
D76E 4F                      MOV     C,A               ;SAVE BAD DATA
D76F 2177D7                  LXI     H,COMER2                ;RETURN
D772 3EF1                    MVI     A,OF1H            ;PHASE I ERROR LITES
D774 C3ABD7                  JMP     LITES2
                  ;
                  ;
                  ;COMMON ERROR OUTPUT ROUTINE
D777 7A           COMER2:    MOV     A,D               ;HIGH ADDRESS
D778 217ED7                  LXI     H,LOADD2                ;RETURN
D77B C3ABD7                  JMP     LITES2
D77E 7B           LOADD2:    MOV     A,E               ;LOW ADDRES TO LITES
D77F 2185D7                  LXI     H,TPAT2           ;RETURN
D782 C3ABD7                  JMP     LITES2
D785 79           TPAT2:     MOV     A,C               ;TEST PATTERN TO LITES
D786 218CD7                  LXI     H,ACTDA2
D789 C3ABD7                  JMP     LITES2
D78C 78           ACTDA2:    MOV     A,B               ;ACTUAL DATA TO LITES
D78D 2100D7                  LXI     H,ENTER2                     ;START OVER
D790 C3ABD7                  JMP     LITES2
                  ;;
                  ;
                  ; PHASE II ERROR
D793 EB           ERR22:     XCHG                      ;SAVE BAD ADDRESS
D794 82                      ADD     D
D795 47                      MOV     B,A
D796 4A                      MOV     C,D
D797 3EF2                    MVI     A,OF2H            ;PHASE II ERROR TO LITES
D799 2177D7                  LXI     H,COMER2                ;RETURN
D79C C3ABD7                  JMP     LITES2
                  ;
                  ;
                  ; PHASE III ERROR
D79F EB           ERR32:     XCHG              ;SAVE BAD ADDRESS
D7A0 83                      ADD     E
D7A1 47                      MOV     B,A
D7A2 4B                      MOV     C,E
D7A3 3EF3                    MVI     A,OF3H            ;PHASE II ERRO TO LITES
D7A5 2177D7                  LXI     H,COMER2          ;RETURN
D7A8 C3ABD7                  JMP     LITES2
                  ;
                  ;
                  ;LITES ROUTINE       ENTER WITH RETURN IN REG H&L
                  ;
D7AB 2F           LITES2:    CMA                         DATA FOR LITES IN A
D7AC D3FF                    OUT     OFFH              ;OUTPUT LITES
D7AE F9                      SPHL                      ;SAVE RETURN IN SP
D7AF DBFF                    IN      OFFH              ;READ SENSE SWITCHES
D7B1 67                      MOV     H,A               ;SAVE IN H
D7B2 DBFF         LP72:      IN      OFFH              ;READ SWITCHES
D7B4 AC                      XRA     H                 ;SEE IF THEY CHANGED
D7B5 CAB2D7                  JZ      LP72
D7B8 2118FC                  LXI     H,OFC18H                ;DELAY LOOP
D7BB 23           LP82:      INX     H
D7BC AF                      XRA     A
D7BD B4                      ORA     H
D7BE C2BBD7                  JNZ     LP82
D7C1 210000                  LXI     H,0               ;ZERO H
D7C4 39                      DAD     SP                ;MOVE RETURN BACK TO H & L
D7C5 E9                      PCHL                      ;RETURN
```

```
                        ;       8Ø8Ø MONITOR V1.Ø

                        ;       PROGRAMMER: C. E. OHME
                        ;                        (415)657-8326


                        ;
                        ;       SYSTEM CONFIGURATION INTERFACE

F6ØØ                    SCP     EQU     ØF6ØØH
F6ØØ                    IOTAB   EQU     SCP
F63Ø                    ADSCS   EQU     SCP+48
F633                    ADSCR   EQU     SCP+51
F636                    ADIOB   EQU     SCP+54
F639                    ADUST   EQU     SCP+57

                        ;       ASCII CHARACTERS

ØØØD                    CR      EQU     ØDH
ØØØA                    LF      EQU     ØAH

FØØØ                            ORG     ØFØØØH

                        ;       EXTERNALLY REFERENCED SUBROUTINE
                        ;       JUMP TABLE

FØØØ C324FØ                     JMP     BEGIN
FØØ3 C3D6FØ                     JMP     CI
FØØ6 C3EØFØ                     JMP     RI
FØØ9 C3D1FØ                     JMP     CO
FØØC C3E5FØ                     JMP     PO
FØØF C3EAFØ                     JMP     LO
FØ12 C3DBFØ                     JMP     CSTS
FØ15 C3ØEF1                     JMP     IOCHK
FØ18 C315F1                     JMP     IOSET
FØ1B C3B1F1                     JMP     MEMCK
FØ1E C31EF1                     JMP     STRNG
FØ21 ØEØØ               REENT:  MVI     C,Ø
FØ23 21ØØØØ                     LXI     H,Ø
FØ24                            ORG     $-2

FØ24 ØEØ1               BEGIN:  MVI     C,1
FØ26 11FF00                     LXI     D,0FFH
FØ29 C3EAF6                     JMP     INITA
FØ2C EB                         XCHG
FØ2D Ø615                       MVI     B,ENDX-EXIT-1
FØ2F 1176F2                     LXI     D,ENDX-1
FØ32 1B                 BG1:    DCX     D
FØ33 1A                         LDAX    D
FØ34 2B                         DCX     H
FØ35 77                         MOV     M,A
FØ36 Ø5                         DCR     B
```

```
FØ37 C232FØ             JNZ      BG1
FØ3A F9                 SPHL
FØ3B CD39F6             CALL     ADUST
FØ3E E5                 PUSH     H
FØ3F 26ØØ               MVI      H,Ø
FØ41 E5                 PUSH     H
FØ42 E5                 PUSH     H
FØ43 E5                 PUSH     H
FØ44 79                 MOV      A,C
FØ45 B7                 ORA      A
FØ46 CA4EFØ             JZ       BG2
FØ49 CD36F6             CALL     ADIOB
FØ4C 36ØØ               MVI      M,Ø
FØ4E 214ØF7    BG2:     LXI      H, TITLE
FØ51 CD1EF1             CALL     STRNG
FØ54 B7                 ORA      A

               ;        COMMAND RETURN POINT

FØ55 D266FØ    CMNDR:   JNC      START

               ;        ERROR RETURN

FØ58 CD33F6    LER:     CALL     ADSCR
FØ5B 11E3FF             LXI      D,EXIT-ENDX-7
FØ5E 19                 DAD      D
FØ5F F9                 SPHL
FØ6Ø 219DFØ             LXI      H,ERM
FØ63 CD1EF1             CALL     STRNG

               ;        INPUT AND EXECUTE NEXT COMMAND

FØ66 FB        START:   EI
FØ67 CD46F1             CALL     CRLF
FØ6A ØE2E               MVI      C,'.'
FØ6C CDD1FØ             CALL     CO
FØ6F CD2FF1             CALL     TI
FØ72 D641               SUI      'A'
FØ74 FA58FØ             JM       LER
FØ77 FE18               CPI      'X'-'A'+1
FØ79 F258FØ             JP       LER
FØ7C 87                 ADD      A
FØ7D 2155FØ             LXI      H,CMNDR
FØ8Ø E5                 PUSH     H
FØ81 219FFØ             LXI      H,TBL
FØ84 16ØØ               MVI      D,Ø
FØ86 5F                 MOV      E,A
FØ87 19                 DAD      D
FØ88 7E                 MOV      A,M
FØ89 23                 INX      H
FØ8A 66                 MOV      H,M
FØ8B 6F                 MOV      L,A
FØ8C ØEØ2               MVI      C,2
FØ8E E9                 PCHL
```

```
FØ8F ØDØA4D4F VERS:   DB      CR,LF,'MONITOR V1.','Ø' OR 8ØH
FØ93 4E49544F
FØ97 52205631
FØ9B 2EBØ
FØ9D ØAAA      ERM:   D3      LF,'*' OR 8ØH

               ;      COMMAND JUMP TABLE

FØ9F 77F2      TBL:   DW      ASSIGN
FØA1 CØF2             DW      BIN
FØA3 A0F3             DW      HEXN
FØA5 1CF3             DW      DISP
FØA7 58FØ             DW      LER
FØA9 3CF3             DW      FILL
FØAB 4CF3             DW      GOTO
FØAD 00F8             DW      HELP
FØAF 58FØ             DW      LER
FØB1 58FØ             DW      LER
FØB3 BCF3             DW      KOPY
FØB5 C6F3             DW      LOAD
FØB7 ØCF4             DW      MOVE
FØB9 1EF4             DW      NULL
FØBB 58FØ             DW      LER
FØBD 58FØ             DW      LER
FØBF 58FØ             DW      LER
FØC1 23F4             DW      READ
FØC3 89F4             DW      SUBS
FØC5 00FD             DW      TEST
FØC7 58FØ             DW      LER
FØC9 58FØ             DW      LER
FØCB B1F4             DW      WRITE
FØCD 3DF5             DW      X

               ;      UTILITY SUBROUTINES

FØCF ØE2Ø      BLK:   MVI     C,' '

FØD1 CDEFFØ    CO:    CALL    IOBR
FØD4 Ø11Ø             DB      1,1ØH

FØD6 CDEFFØ    CI:    CALL    IOBR
FØD9 Ø1Ø8             DB      1,8

FØDB CDEFFØ    CSTS:  CALL    IOBR
FØDE Ø1ØØ             DB      1,Ø

FØEØ CDEFFØ    RI:    CALL    IOBR
FØE3 Ø418             DB      4,18H

FØE5 CDEFFØ    PO:    CALL    IOBR
FØE8 Ø32Ø             DB      3,2ØH
```

```
FØEA CDEFFØ   LO:      CALL    IOBR
FØED Ø228              DB      2,28H

FØEF E3       IOBR:    XTHL
FØFØ C5                PUSH    B
FØF1 46                MOV     B,M
FØF2 23                INX     H
FØF3 4E                MOV     C,M
FØF4 CD36F6            CALL    ADIOB
FØF7 7E                MOV     A,M
FØF8 ØF                RRC
FØF9 Ø7       IOB1:    RLC
FØFA Ø7                RLC
FØFB Ø5                DCR     B
FØFC C2F9FØ            JNZ     IOB1
FØFF E6Ø6              ANI     6
F1Ø1 81                ADD     C
F1Ø2 4F                MOV     C,A
F1Ø3 21ØØF6            LXI     H,IOTAB
F1Ø6 Ø9                DAD     B
F1Ø7 7E                MOV     A,M
F1Ø8 23                INX     H
F1Ø9 66                MOV     H,M
F1ØA 6F                MOV     L,A
F1ØB C1                POP     B
F1ØC E3                XTHL
F1ØD C9                RET
              IOCHK:
F1ØE E5                PUSH    H
F1ØF CD36F6            CALL    ADIOB
F112 7E                MOV     A,M
F113 E1                POP     H
F114 C9                RET
              IOSET:
F115 E5                PUSH    H
F116 F5                PUSH    PSW
F117 CD36F6            CALL    ADIOB
F11A 71                MOV     M,C
F11B F1                POP     PSW
F11C E1                POP     H
F11D C9                RET
              STRNG:
F11E 7E                MOV     A,M
F11F E67F              ANI     7FH
F121 C8                RZ
F122 4F                MOV     C,A
F123 7E                MOV     A,M
F124 B7                ORA     A
F125 FAD1FØ            JM      CO
F128 CDD1FØ            CALL    CO
F12B 23                INX     H
F12C C31EF1            JMP     STRNG
```

```
                    TI:
F12F CDD6FØ                   CALL    CI
F132 E67F                     ANI     7FH
F134 C5                       PUSH    B
F135 4F                       MOV     C,A
F136 CDD1FØ                   CALL    CO
F139 79                       MOV     A,C
F13A C1                       POP     B
F13B C9                       RET
                    CONV:
F13C E6ØF                     ANI     ØFH
F13E C69Ø                     ADI     9ØH
F14Ø 27                       DAA
F141 CE4Ø                     ACI     4ØH
F143 27                       DAA
F144 4F                       MOV     C,A
F145 C9                       RET
                    CRLF:
F146 ØEØD                     MVI     C,CR
F148 CDD1FØ                   CALL    CO
F14B ØEØA                     MVI     C,LF
F14D C3D1FØ                   JMP     CO
                    EXPR1:
F15Ø ØEØ1                     MVI     C,1
                    EXPR:
F152 21ØØØØ                   LXI     H,ØØØØH
F155 CD2FF1         EXØ:      CALL    TI
F158 47             EX1:      MOV     B,A
F159 CDDØF1                   CALL    NIBBL
F15C DA68F1                   JC      EX2
F15F 29                       DAD     H
F16Ø 29                       DAD     H
F161 29                       DAD     H
F162 29                       DAD     H
F163 B5                       ORA     L
F164 6F                       MOV     L,A
F165 C355F1                   JMP     EXØ
F168 E3             EX2:      XTHL
F169 E5                       PUSH    H
F16A 78                       MOV     A,B
F16B CDE5F1                   CALL    P2C
F16E D276F1                   JNC     EX3
F171 ØD                       DCR     C
F172 C258FØ                   JNZ     LER
F175 C9                       RET
F176 C258FØ         EX3:      JNZ     LER
F179 ØD                       DCR     C
F17A C252F1                   JNZ     EXPR
F17D C9                       RET
                    EXF:
F17E ØEØ1                     MVI     C,1
F18Ø 21ØØØØ                   LXI     H,ØØØØH
F183 C358F1                   JMP     EX1
```

```
                   HILO:
F186 23                      INX     H
F187 7C                      MOV     A,H
F188 B5                      ORA     L
F189 37                      STC
F18A C8                      RZ
F18B 7B                      MOV     A,E
F18C 95                      SUB     L
F18D 7A                      MOV     A,D
F18E 9C                      SBB     H
F18F C9                      RET
                   LADR:
F19Ø 7C                      MOV     A,H
F191 CD95F1                  CALL    LBYTE
F194 7D                      MOV     A,L
                   LBYTE:
F195 F5                      PUSH    PSW
F196 ØF                      RRC
F197 ØF                      RRC
F198 ØF                      RRC
F199 ØF                      RRC
F19A CD9EF1                  CALL    HXD
F19D F1                      POP     PSW
                   HXD:
F19E CD3CF1                  CALL    CONV
F1A1 C3D1FØ                  JMP     CO
                   LEADS:
F1A4 Ø6Ø4                    MVI     B,4
                   LEAD:
F1A6 ØEØØ                    MVI     C,Ø
F1A8 CDE5FØ                  CALL    PO
F1AB Ø5                      DCR     B
F1AC C2A6F1                  JNZ     LEAD
F1AF B7                      ORA     A
F1BØ C9                      RET
                   MEMCK:
F1B1 E5                      PUSH    H
F1B2 D5                      PUSH    D
F1B3 CD33F6                  CALL    ADSCR
F1B6 EB                      XCHG
F1B7 21ØØØØ                  LXI     H,Ø
F1BA 24            MEMØ:     INR     H
F1BB 7E                      MOV     A,M
F1BC 2F                      CMA
F1BD 77                      MOV     M,A
F1BE BE                      CMP     M
F1BF 2F                      CMA
F1CØ 77                      MOV     M,A
F1C1 CABAF1                  JZ      MEMØ
F1C4 2B                      DCX     H
F1C5 44                      MOV     B,H
F1C6 7C                      MOV     A,H
F1C7 BA                      CMP     D
```

```
F1C8 3ECØ                   MVI     A,ØCØH
F1CA D1                     POP     D
F1CB E1                     POP     H
F1CC C8                     RZ
F1CD 3EFF                   MVI     A,ØFFH
F1CF C9                     RET
                  NIBBL:
F1DØ D63Ø                   SUI     'Ø'
F1D2 D8                     RC
F1D3 C6E9                   ADI     'Ø'-'G'
F1D5 D8                     RC
F1D6 C6Ø6                   ADI     6
F1D8 F2DEF1                 JP      NIØ
F1DB C6Ø7                   ADI     7
F1DD D8                     RC
F1DE C6ØA         NIØ:      ADI     1Ø
F1EØ B7                     ORA     A
F1E1 C9                     RET
                  PCHK:
F1E2 CD2FF1                 CALL    TI
                  P2C:
F1E5 FE2Ø                   CPI     ' '
F1E7 C8                     RZ
F1E8 FE2C                   CPI     ','
F1EA C8                     RZ
F1EB FEØD                   CPI     CR
F1ED 37                     STC
F1EE C8                     RZ
F1EF 3F                     CMC
F1FØ C9                     RET

                  ;         BREAKPOINT ENTRY POINT

F1F1 E5           RESTRT:   PUSH    H
F1F2 D5                     PUSH    D
F1F3 C5                     PUSH    B
F1F4 F5                     PUSH    PSW
F1F5 CD33F6                 CALL    ADSCR
F1F8 11EBFF                 LXI     D,EXIT-ENDX+1
F1FB 19                     DAD     D
F1FC EB                     XCHG
F1FD 21ØAØØ                 LXI     H,ØØØAH
F2ØØ 39                     DAD     SP
F2Ø1 Ø6Ø4                   MVI     B,4
F2Ø3 EB                     XCHG
F2Ø4 2B           RSTØ:     DCX     H
F2Ø5 72                     MOV     M,D
F2Ø6 2B                     DCX     H
F2Ø7 73                     MOV     M,E
F2Ø8 D1                     POP     D
F2Ø9 Ø5                     DCR     B
F2ØA C2Ø4F2                 JNZ     RSTØ
F2ØD C1                     POP     B
```

```
F2ØE ØB                 DCX     B
F2ØF F9                 SPHL
F21Ø 2114ØØ             LXI     H,TLOC
F213 39                 DAD     SP
F214 7E                 MOV     A,M
F215 91                 SUB     C
F216 23                 INX     H
F217 C21FF2             JNZ     RST1
F21A 7E                 MOV     A,M
F21B 9Ø                 SUB     B
F21C CA2DF2             JZ      RST3
F21F 23         RST1:   INX     H
F22Ø 23                 INX     H
F221 7E                 MOV     A,M
F222 91                 SUB     C
F223 C22CF2             JNZ     RST2
F226 23                 INX     H
F227 7E                 MOV     A,M
F228 9Ø                 SUB     B
F229 CA2DF2             JZ      RST3
F22C Ø3         RST2:   INX     B
F22D 21ØFØØ     RST3:   LXI     H,LLOC
F23Ø 39                 DAD     SP
F231 73                 MOV     M,E
F232 23                 INX     H
F233 72                 MOV     M,D
F234 23                 INX     H
F235 23                 INX     H
F236 71                 MOV     M,C
F237 23                 INX     H
F238 7Ø                 MOV     M,B
F239 C5                 PUSH    B
F23A 219DFØ             LXI     H,ERM
F23D CD1EF1             CALL    STRNG
F24Ø E1                 POP     H
F241 CD9ØF1             CALL    LADR
F244 2114ØØ             LXI     H,TLOC
F247 39                 DAD     SP
F248 16Ø2               MVI     D,2
F24A 4E         RST4:   MOV     C,M
F24B 36ØØ               MVI     M,Ø
F24D 23                 INX     H
F24E 46                 MOV     B,M
F24F 36ØØ               MVI     M,Ø
F251 23                 INX     H
F252 79                 MOV     A,C
F253 BØ                 ORA     B
F254 CA59F2             JZ      RST5
F257 7E                 MOV     A,M
F258 Ø2                 STAX    B
F259 23         RST5:   INX     H
F25A 15                 DCR     D
F25B C24AF2             JNZ     RST4
F25E C366FØ             JMP     START
```

```
                      ;       SCRATCHPAD TEMPLATE

F261 D1               EXIT:   POP      D
F262 C1                       POP      B
F263 F1                       POP      PSW
F264 E1                       POP      H
F265 F9                       SPHL
F266 FB                       EI
F267 21ØØØØ                   LXI      H,Ø
F268                  HLX     EQU      $-2
F26A C3ØØØØ                   JMP      Ø
F26B                  PCX     EQU      $-2
F26D ØØØØ             TIA:    DW       Ø        ;TRAP 1 ADDR
F26F ØØ                       DB       Ø        ;TRAP 1 INST
F27Ø ØØØØ                     DW       Ø        ;TRAP 2 ADDR
F272 ØØ                       DB       Ø        ;TRAP 2 INST
F273 ØØØØ                     DW       Ø        ;VIDEO PNTR
F275 ØØ                       DB       Ø        ;VIDEO HOLD
F276 ØØ                       DB       Ø        ;IOBYT
                      ENDX:
ØØØ5                  ALOC    EQU      5
ØØØ3                  BLOC    EQU      3
ØØØ2                  CLOC    EQU      2
ØØØ1                  DLOC    EQU      1
ØØØØ                  ELOC    EQU      Ø
ØØØ4                  FLOC    EQU      4
ØØ1Ø                  HLOC    EQU      HLX-EXIT+9
ØØØF                  LLOC    EQU      HLX-EXIT+8
ØØ13                  PLOC    EQU      PCX-EXIT+9
ØØØ7                  SLOC    EQU      7
ØØ14                  TLOC    EQU      TIA-EXIT+8

                      ;       COMMAND IMPLEMENTATION

                      ;       ASSIGN COMMAND

F277 CD2FF1           ASSIGN: CALL     TI
F27A Ø6ØØ                     MVI      B,Ø
F27C FE43                     CPI      'C'
F27E CA93F2                   JZ       AS1
F281 Ø4                       INR      B
F282 FE52                     CPI      'R'
F284 CA93F2                   JZ       AS1
F287 Ø4                       INR      B
F288 FE5Ø                     CPI      'P'
F28A CA93F2                   JZ       AS1
F28D Ø4                       INR      B
F28E FE4C                     CPI      'L'
F29Ø C2BEF2                   JNZ      EREXT
F293 CD2FF1           AS1:    CALL     TI
F296 FE3D                     CPI      '='
F298 C293F2                   JNZ      AS1
```

```
F29B CD2FF1             CALL    TI
F29E D630               SUI     '0'
F2A0 6F                 MOV     L,A
F2A1 FABEF2             JM      EREXT
F2A4 FE04               CPI     4
F2A6 F2BEF2             JP      EREXT
F2A9 2603               MVI     H,3
F2AB 05         AS2:    DCR     B
F2AC FAB4F2             JM      AS3
F2AF 29                 DAD     H
F2B0 29                 DAD     H
F2B1 C3ABF2             JMP     AS2
F2B4 EB         AS3:    XCHG
F2B5 CD36F6             CALL    ADIOB
F2B8 7E                 MOV     A,M
F2B9 B2                 ORA     D
F2BA AA                 XRA     D
F2BB B3                 ORA     E
F2BC 77                 MOV     M,A
F2BD C9                 RET
F2BE 37         EREXT:  STC
F2BF C9                 RET

                ;       BINARY COMMAND

F2C0 CD52F1     BIN:    CALL    EXPR
F2C3 CD46F1             CALL    CRLF
F2C6 D1                 POP     D
F2C7 E1                 POP     H
F2C8 7A         BIN0:   MOV     A,D
F2C9 B3                 ORA     E
F2CA C2D7F2             JNZ     B0
F2CD CDA4F1             CALL    LEADS
F2D0 0E78               MVI     C,78H
F2D2 CD11F3             CALL    PHL
F2D5 B7                 ORA     A
F2D6 C9                 RET
F2D7 7B         B0:     MOV     A,E
F2D8 95                 SUB     L
F2D9 7A                 MOV     A,D
F2DA 9C                 SBB     H
F2DB D8                 RC
F2DC 7B         B1:     MOV     A,E
F2DD 95                 SUB     L
F2DE 4F                 MOV     C,A
F2DF 7A                 MOV     A,D
F2E0 9C                 SBB     H
F2E1 3F                 CMC
F2E2 D0                 RNC
F2E3 0C                 INR     C
F2E4 C2E9F2             JNZ     B2
F2E7 0EFF               MVI     C,0FFH
F2E9 D5         B2:     PUSH    D
```

```
F2EA 59                 MOV     E,C
F2EB CDA4F1             CALL    LEADS
F2EE 0E3C               MVI     C,3CH
F2F0 CDE5F0             CALL    PO
F2F3 4B                 MOV     C,E
F2F4 CD11F3             CALL    PHL
F2F7 7C                 MOV     A,H
F2F8 85                 ADD     L
F2F9 57                 MOV     D,A
F2FA 4E         B3:     MOV     C,M
F2FB 23                 INX     H
F2FC 7A                 MOV     A,D
F2FD 81                 ADD     C
F2FE 57                 MOV     D,A
F2FF CDE5F0             CALL    PO
F302 1D                 DCR     E
F303 C2FAF2             JNZ     B3
F306 4A                 MOV     C,D
F307 CDE5F0             CALL    PO
F30A D1                 POP     D
F30B 7D                 MOV     A,L
F30C B4                 ORA     H
F30D C8                 RZ
F30E C3DCF2             JMP     B1
F311 CDE5F0     PHL:    CALL    PO
F314 4D                 MOV     C,L
F315 CDE5F0             CALL    PO
F318 4C                 MOV     C,H
F319 C3E5F0             JMP     PO

                ;       DISPLAY COMMAND

F31C CD52F1     DISP:   CALL    EXPR
F31F D1                 POP     D
F320 E1                 POP     H
F321 CD46F1     DI0:    CALL    CRLF
F324 CD90F1             CALL    LADR
F327 CDCFF0     DI1:    CALL    BLK
F32A 7E                 MOV     A,M
F32B CD95F1             CALL    LBYTE
F32E CD86F1             CALL    HILO
F331 3F                 CMC
F332 D0                 RNC
F333 7D                 MOV     A,L
F334 E60F               ANI     0FH
F336 C227F3             JNZ     DI1
F339 C321F3             JMP     DI0

                ;       FILL COMMAND

F33C 0C         FILL:   INR     C
F33D CD52F1             CALL    EXPR
F340 C1                 POP     B
```

```
F341 D1                POP     D
F342 E1                POP     H
F343 71         F1Ø:   MOV     M,C
F344 CD86F1            CALL    HILO
F347 D243F3            JNC     F1Ø
F34A B7                ORA     A
F34B C9                RET

                ;      GOTO COMMAND

F34C E1         GOTO:  POP     H
F34D CDE2F1            CALL    PCHK
F35Ø DA98F3            JC      GO3
F353 CA72F3            JZ      GOØ
F356 CD7EF1            CALL    EXF
F359 D1                POP     D
F35A 2113ØØ            LXI     H,PLOC
F35D 39                DAD     SP
F35E 72                MOV     M,D
F35F 2B                DCX     H
F36Ø 73                MOV     M,E
F361 78                MOV     A,B
F362 FEØD              CPI     CR
F364 CA98F3            JZ      GO3
F367 3EC3              MVI     A,(JMP Ø)
F369 32Ø8ØØ            STA     8
F36C 21F1F1            LXI     H,RESTRT
F36F 22Ø9ØØ            SHLD    9
F372 16Ø2       GOØ:   MVI     D,2
F374 2114ØØ            LXI     H,TLOC
F377 39                DAD     SP
F378 E5         GO1:   PUSH    H
F379 CD5ØF1            CALL    EXPR1
F37C 58                MOV     E,B
F37D C1                POP     B
F37E E1                POP     H
F37F 78                MOV     A,B
F38Ø B1                ORA     C
F381 CA8EF3            JZ      GO2
F384 71                MOV     M,C
F385 23                INX     H
F386 7Ø                MOV     M,B
F387 23                INX     H
F388 ØA                LDAX    B
F389 77                MOV     M,A
F38A 23                INX     H
F38B 3ECF              MVI     A,(RST 1)
F38D Ø2                STAX    B
F38E 7B         GO2:   MOV     A,E
F38F FEØD              CPI     CR
F391 CA98F3            JZ      GO3
F394 15                DCR     D
F395 C278F3            JNZ     GO1
```

```
F398 CD46F1    GO3:     CALL     CRLF
F39B 21Ø8ØØ             LXI      H,ØØØ8H
F39E 39                 DAD      SP
F39F E9                 PCHL

               ;        HEXADECIMAL COMMAND

F3AØ CD52F1    HEXN:    CALL     EXPR
F3A3 D1                 POP      D
F3A4 E1                 POP      H
F3A5 CD46F1             CALL     CRLF
F3A8 E5                 PUSH     H
F3A9 19                 DAD      D
F3AA CD9ØF1             CALL     LADR
F3AD CDCFFØ             CALL     BLK
F3BØ E1                 POP      H
F3B1 7D                 MOV      A,L
F3B2 93                 SUB      E
F3B3 6F                 MOV      L,A
F3B4 7C                 MOV      A,H
F3B5 9A                 SBB      D
F3B6 67                 MOV      H,A
F3B7 CD9ØF1             CALL     LADR
F3BA B7                 ORA      A
F3BB C9                 RET

               ;        COPY COMMAND

F3BC CDEØFØ    KOPY:    CALL     RI
F3BF 4F                 MOV      C,A
F3CØ CDE5FØ             CALL     PO
F3C3 C3BCF3             JMP      KOPY

               ;        LOAD COMMAND

F3C6 CD5ØF1    LOAD:    CALL     EXPR1
F3C9 C1                 POP      B
F3CA CDEØFØ    L1:      CALL     RI
F3CD D8                 RC
F3CE FE3C               CPI      3CH
F3DØ CADFF3             JZ       L2
F3D3 FE78               CPI      78H
F3D5 C2CAF3             JNZ      L1
F3D8 CDØ1F4             CALL     LHL
F3DB D8                 RC
F3DC B5                 ORA      L
F3DD C8                 RZ
F3DE E9                 PCHL
F3DF CDEØFØ    L2:      CALL     RI
F3E2 D8                 RC
F3E3 5F                 MOV      E,A
F3E4 CDØ1F4             CALL     LHL
F3E7 D8                 RC
```

```
F3E8 85                   ADD     L
F3E9 57                   MOV     D,A
F3EA Ø9                   DAD     B
F3EB CDEØFØ     L3:       CALL    RI
F3EE D8                   RC
F3EF 77                   MOV     M,A
F3FØ 82                   ADD     D
F3F1 57                   MOV     D,A
F3F2 23                   INX     H
F3F3 1D                   DCR     E
F3F4 C2EBF3               JNZ     L3
F3F7 CDEØFØ               CALL    RI
F3FA D8                   RC
F3FB BA                   CMP     D
F3FC CACAF3               JZ      L1
F3FF 37                   STC
F4ØØ C9                   RET
F4Ø1 CDEØFØ     LHL:      CALL    RI
F4Ø4 D8                   RC
F4Ø5 6F                   MOV     L,A
F4Ø6 CDEØFØ               CALL    RI
F4Ø9 D8                   RC
F4ØA 67                   MOV     H,A
F4ØB C9                   RET

                ;         MOVE COMMAND

F4ØC ØC         MOVE:     INR     C
F4ØD CD52F1               CALL    EXPR
F41Ø C1                   POP     B
F411 D1                   POP     D
F412 E1                   POP     H
F413 7E         MVØ:      MOV     A,M
F414 Ø2                   STAX    B
F415 Ø3                   INX     B
F416 CD86F1               CALL    HILO
F419 D213F4               JNC     MVØ
F41C B7                   ORA     A
F41D C9                   RET

                ;         NULL COMMAND

F41E Ø63C       NULL:     MVI     B,6Ø
F42Ø C3A6F1               JMP     LEAD

                ;         READ COMMAND

F423 CD5ØF1     READ:     CALL    EXPR1
F426 E1                   POP     H
F427 CDEØFØ     REDØ:     CALL    RI
F42A D8                   RC
F42B E67F                 ANI     7FH
F42D D63A                 SUI     ':'
```

```
F42F C227F4           JNZ    REDØ
F432 57               MOV    D,A
F433 E5               PUSH   H
F434 CD65F4           CALL   BYTE
F437 CA59F4           JZ     RED2
F43A 5F               MOV    E,A
F43B CD65F4           CALL   BYTE
F43E 47               MOV    B,A
F43F CD65F4           CALL   BYTE
F442 4F               MOV    C,A
F443 Ø9               DAD    B
F444 CD65F4           CALL   BYTE
F447 CD65F4   RED1:   CALL   BYTE
F44A 77               MOV    M,A
F44B 23               INX    H
F44C 1D               DCR    E
F44D C247F4           JNZ    RED1
F45Ø CD65F4           CALL   BYTE
F453 E1               POP    H
F454 CA27F4           JZ     REDØ
F457 37               STC
F458 C9               RET
F459 CD65F4   RED2:   CALL   BYTE
F45C 67               MOV    H,A
F45D CD65F4           CALL   BYTE
F46Ø C1               POP    B
F461 6F               MOV    L,A
F462 B4               ORA    H
F463 C8               RZ
F464 E9               PCHL
F465 CD76F4   BYTE:   CALL   RNBBL
F468 Ø7               RLC
F469 Ø7               RLC
F46A Ø7               RLC
F46B Ø7               RLC
F46C 4F               MOV    C,A
F46D CD76F4           CALL   RNBBL
F47Ø B1               ORA    C
F471 4F               MOV    C,A
F472 82               ADD    D
F473 57               MOV    D,A
F474 79               MOV    A,C
F475 C9               RET
F476 CDEØFØ   RNBBL:  CALL   RI
F479 DA85F4           JC     RNBER
F47C E67F             ANI    7FH
F47E CDDØF1           CALL   NIBBL
F481 DA85F4           JC     RNBER
F484 C9               RET
F485 E1       RNBER:  POP    H
F486 E1               POP    H
F487 E1               POP    H
F488 C9               RET
```

```
                    ;           SUBSTITUTE COMMAND

F489  CD5ØF1     SUBS:      CALL      EXPR1
F48C  CDE5F1                CALL      P2C
F48F  E1                    POP       H
F49Ø  D8                    RC
F491  7E         SUØ:       MOV       A,M
F492  CD95F1                CALL      LBYTE
F495  ØE2D                  MVI       C,'-'
F497  CDD1FØ                CALL      CO
F49A  CDE2F1                CALL      PCHK
F49D  3F                    CMC
F49E  DØ                    RNC
F49F  CAADF4                JZ        SU1
F4A2  E5                    PUSH      H
F4A3  CD7EF1                CALL      EXF
F4A6  D1                    POP       D
F4A7  E1                    POP       H
F4A8  73                    MOV       M,E
F4A9  78                    MOV       A,B
F4AA  FEØD                  CPI       CR
F4AC  C8                    RZ
F4AD  23         SU1:       INX       H
F4AE  C391F4                JMP       SUØ

                    ;           WRITE COMMAND

F4B1  CD52F1     WRITE:     CALL      EXPR
F4B4  CD46F1                CALL      CRLF
F4B7  D1                    POP       D
F4B8  E1                    POP       H
F4B9  7A         WRITØ:     MOV       A,D
F4BA  B3                    ORA       E
F4BB  C2DBF4                JNZ       WØ
F4BE  CD33F5                CALL      PEOL
F4C1  ØE3A                  MVI       C,':'
F4C3  CDE5FØ                CALL      PO
F4C6  AF                    XRA       A
F4C7  57                    MOV       D,A
F4C8  CD1CF5                CALL      PBYTE
F4CB  CD17F5                CALL      PADR
F4CE  3EØ1                  MVI       A,1
F4DØ  CD1CF5                CALL      PBYTE
F4D3  AF                    XRA       A
F4D4  92                    SUB       D
F4D5  CD1CF5                CALL      PBYTE
F4D8  C31EF4                JMP       NULL
F4DB  7B         WØ:        MOV       A,E
F4DC  95                    SUB       L
F4DD  7A                    MOV       A,D
F4DE  9C                    SBB       H
F4DF  D8                    RC
```

```
F4EØ 7B         WRIØ:   MOV     A,E
F4E1 95                 SUB     L
F4E2 4F                 MOV     C,A
F4E3 7A                 MOV     A,D
F4E4 9C                 SBB     H
F4E5 3F                 CMC
F4E6 DØ                 RNC
F4E7 79                 MOV     A,C
F4E8 E6ØF               ANI     ØFH
F4EA 3C                 INR     A
F4EB D5                 PUSH    D
F4EC 5F                 MOV     E,A
F4ED 16ØØ               MVI     D,Ø
F4EF CD33F5             CALL    PEOL
F4F2 ØE3A               MVI     C,':'
F4F4 CDE5FØ             CALL    PO
F4F7 7B                 MOV     A,E
F4F8 CD1CF5             CALL    PBYTE
F4FB CD17F5             CALL    PADR
F4FE AF                 XRA     A
F4FF CD1CF5             CALL    PBYTE
F5Ø2 7E         WRI3:   MOV     A,M
F5Ø3 23                 INX     H
F5Ø4 CD1CF5             CALL    PBYTE
F5Ø7 1D                 DCR     E
F5Ø8 C2Ø2F5             JNZ     WRI3
F5ØB AF                 XRA     A
F5ØC 92                 SUB     D
F5ØD CD1CF5             CALL    PBYTE
F51Ø D1                 POP     D
F511 7D                 MOV     A,L
F512 B4                 ORA     H
F513 C8                 RZ
F514 C3EØF4             JMP     WRIØ
F517 7C         PADR:   MOV     A,H
F518 CD1CF5             CALL    PBYTE
F51B 7D                 MOV     A,L
F51C F5         PBYTE:  PUSH    PSW
F51D ØF                 RRC
F51E ØF                 RRC
F51F ØF                 RRC
F52Ø ØF                 RRC
F521 CD3CF1             CALL    CONV
F524 CDE5FØ             CALL    PO
F527 F1                 POP     PSW
F528 F5                 PUSH    PSW
F529 CD3CF1             CALL    CONV
F52C CDE5FØ             CALL    PO
F52F F1                 POP     PSW
F53Ø 82                 ADD     D
F531 57                 MOV     D,A
F532 C9                 RET
F533 ØEØD       PEOL:   MVI     C,CR
```

```
F535 CDE5FØ           CALL    PO
F538 ØEØA             MVI     C,LF
F53A C3E5FØ           JMP     PO

                ;     REGISTER COMMAND

F53D CD2FF1     X:    CALL    TI
F54Ø 21CCF5           LXI     H,ACTBL
F543 FEØD             CPI     CR
F545 CA9FF5           JZ      X6
F548 47               MOV     B,A
F549 BE         XØ:   CMP     M
F54A CA57F5           JZ      X1
F54D 7E               MOV     A,M
F54E 17               RAL
F54F D8               RC
F55Ø 23               INX     H
F551 23               INX     H
F552 23               INX     H
F553 78               MOV     A,B
F554 C349F5           JMP     XØ
F557 CDCFFØ     X1:   CALL    BLK
F55A 23         X2:   INX     H
F55B 7E               MOV     A,M
F55C EB               XCHG
F55D 6F               MOV     L,A
F55E 26ØØ             MVI     H,Ø
F56Ø 39               DAD     SP
F561 EB               XCHG
F562 23               INX     H
F563 46               MOV     B,M
F564 23               INX     H
F565 1A               LDAX    D
F566 CD95F1           CALL    LBYTE
F569 Ø5               DCR     B
F56A CA72F5           JZ      X3
F56D 1B               DCX     D
F56E 1A               LDAX    D
F56F CD95F1           CALL    LBYTE
F572 Ø4         X3:   INR     B
F573 ØE2D             MVI     C,'-'
F575 CDD1FØ           CALL    CO
F578 CDE2F1           CALI.   PCHK
F57B 3F               CMC
F57C DØ               RNC
F57D CA95F5           JZ      X5
F58Ø E5               PUSH    H
F581 C5               PUSH    B
F582 CD7EF1           CALL    EXF
F585 E1               POP     H
F586 F1               POP     PSW
F587 C5               PUSH    B
F588 F5               PUSH    PSW
```

```
F589 7D                 MOV    A,L
F58A 12                 STAX   D
F58B C1                 POP    B
F58C Ø5                 DCR    B
F58D CA93F5             JZ     X4
F59Ø 13                 INX    D
F591 7C                 MOV    A,H
F592 12                 STAX   D
F593 C1         X4:     POP    B
F594 E1                 POP    H
F595 7E         X5:     MOV    A,M
F596 B7                 ORA    A
F597 F8                 RM
F598 78                 MOV    A,B
F599 FEØD               CPI    CR
F59B C8                 RZ
F59C C35AF5             JMP    X2
F59F CD46F1     X6:     CALL   CRLF
F5A2 CDCFFØ     X7:     CALL   BLK
F5A5 7E                 MOV    A,M
F5A6 23                 INX    H
F5A7 B7                 ORA    A
F5A8 F8                 RM
F5A9 4F                 MOV    C,A
F5AA CDD1FØ             CALL   CO
F5AD ØE3D               MVI    C,'='
F5AF CDD1FØ             CALL   CO
F5B2 7E                 MOV    A,M
F5B3 23                 INX    H
F5B4 EB                 XCHG
F5B5 6F                 MOV    L,A
F5B6 26ØØ               MVI    H,Ø
F5B8 39                 DAD    SP
F5B9 EB                 XCHG
F5BA 46                 MOV    B,M
F5BB 23                 INX    H
F5BC 1A                 LDAX   D
F5BD CD95F1             CALL   LBYTE
F5CØ Ø5                 DCR    B
F5C1 CAA2F5             JZ     X7
F5C4 1B                 DCX    D
F5C5 1A                 LDAX   D
F5C6 CD95F1             CALL   LBYTE
F5C9 C3A2F5             JMP    X7
F5CC 41Ø7Ø1     ACTBL:  DB     'A',     ALOC+2,  1
F5CF 42Ø5Ø1             DB     'B',     BLOC+2;  1
F5D2 43Ø4Ø1             DB     'C',     CLOC+2,  1
F5D5 44Ø3Ø1             DB     'D',     DLOC+2,  1
F5D8 45Ø2Ø1             DB     'E',     ELOC+2,  1
F5DB 46Ø6Ø1             DB     'F',     FLOC+2,  1
F5DE 4812Ø1             DB     'H',     HLOC+2,  1
```

```
F5E1 4C1101              DB       'L',     LLOC+2, 1
F5E4 4D1202              DB       'M',     MLOC+2, 2
F5E7 501502              DB       'P',     PLOC+2, 2
F5EA 530902              DB       'S',     SLOC+2, 2
F5ED FF                  DB       -1

0000                     END
```

```
                    ;       SYSTEM CONFIGURATION PACKAGE

F6ØØ                        ORG     ØF6ØØH

                    ;       LOGICAL DEVICE/DEVICE DRIVER TABLES
                    ;
                    ;       EACH 4 ENTRY TABLE LISTS THE ADDRESSES
                    ;       OF THE DRIVER ROUTINES TO BE USED FOR
                    ;       THE PHYSICAL DEVICES WHICH MAY ASSIGNED
                    ;       TO THAT LOGICAL DEVICE.

                    IOTAB:

                    ;       CONSOLE STATUS

                    ;       RETURN WITH REGISTER A = Ø IF NO
                    ;       CONSOLE CHARACTER AVAILABLE.

F6ØØ AØF6           CSTAB:  DW      TTST    ;Ø
F6Ø2 7FF6                   DW      KYST    ;1
F6Ø4 7FF6                   DW      KYST    ;2
F6Ø6 7FF6                   DW      KYST    ;3

                    ;       CONSOLE INPUT

                    ;       RETURN CONSOLE INPUT CHARACTER
                    ;       IN REGISTER A.

F6Ø8 A8F6           CITAB:  DW      TTI     ;Ø
F6ØA 65F6                   DW      KYBD    ;1
F6ØC 66F6                   DW      XYBD    ;2
F6ØE 66F6                   DW      KYBD    ;3

                    ;       CONSOLE OUTPUT

                    ;       OUTPUT BYTE IN REGISTER C
                    ;       TO CONSOLE OUTPUT DEVICE.

F61Ø B7F6           COTAB:  DW      TTO     ;Ø
F612 B7F6                   DW      TTO     ;1
F614 D4F6                   DW      THRM    ;2
F616 59F6                   DW      CRT     ;3

                    ;       READER INPUT

                    ;       RETURN READER INPUT BYTE IN
                    ;       REGISTER A, CARRY OFF.  SET
                    ;       CARRY IF NO BYTE AVAILABLE.
```

```
F618 C2F6      RITAB:  DW      TTR      ;Ø
F61A 87F6              DW      RDR      ;1
F61C 66F6              DW      KYBD     ;2
F61E FØB8              DW      ØB8FØH   ;3 DISK READ

               ;       PUNCH OUTPUT

               ;       OUTPUT BYTE IN REGISTER C
               ;       TO PUNCH DEVICE.

F62Ø B7F6      POTAB:  DW      TTO      ;Ø
F622 DFF6              DW      PUNCH    ;1
F624 59F6              DW      CRT      ;2
F626 73B9              DW      ØB973H   ;3 DISK WRITE

               ;       LISTING OUTPUT

               ;       OUTPUT BYTE IN REGISTER C
               ;       TO LISTING DEVICE.

F628 B7F6      LOTAB:  DW      TTO      ;Ø
F62A 59F6              DW      CRT      ;1
F62C D4F6              DW      THRM     ;2
F62E B7F6              DW      TTO      ;3


               ;       SPECIAL SUBROUTINE TO LOCATE MONITOR
               ;       SCRATCH RAM
               ;
               ;       THE ADDRESS OF THE TOP OF THE SCRATCH
               ;       RAM AREA USED BY THE MONITOR IS RETURNED
               ;       IN REGISTERS D,E.
               ;NOTE:  THIS SUBROUTINE IS NOT CALLED IN THE
               ;       USUAL WAY:  INSTEAD, THE RETURN ADDRESS
               ;       IS PLACED IN REGISTERS D,E AND THE
               ;       SUBROUTINE IS ENTERED BY A JUMP INSTRUCTI
               ;       RETURN IS DONE BY PLACING THE RETURN
               ;       ADDRESS IN H,L AND EXECUTING A PCHL INST.

F63Ø C33DF6    ADSCS:  JMP     ADS2
```

```
F633 C34FF6   ADSCR:  JMP     ADS1

                ;       SUBROUTINE TO SET ADDRESS
                ;       OF IOBYT
                ;
                ;       THE ADDRESS OF THE BYTE USED TO
                ;       RECORD THE CURRENT PHYSICAL/LOGICAL
                ;       DEVICE ASSIGNMENTS IS RETURNED IN
                ;       REGISTERS H,L.

F636 C34FF6   ADIOB:  JMP     ADS1


                ;       SUBROUTINE TO SET THE USER STACK
                ;       ADDRESS.
                ;
                ;       THE ADDRESS TO BE USED AS THE
                ;       DEFAULT VALUE OF THE USER STACK
                ;       ADDRESS IS RETURNED IN REGISTERS H,L.

F639 218000   ADUST:  LXI     H,80H
F63C C9               RET

F63D 21ØØØØ   ADS2:   LXI     H,Ø
F64Ø 24       ADS3:   INR     H
F641 7E               MOV     A,M
F642 2F               CMA
F643 F3               DI
F644 77               MOV     M,A
F645 BE               CMP     M
F646 2F               CMA
F647 FB               EI
F648 77               MOV     M,A
F649 CA4ØF6           JZ      ADS3
F64C 2B               DCX     H,
F64D EB               XCHG
F64E E9               PCHL

F64F D5       ADS1:   PUSH    D
F65Ø 12FF00           LXI     H,OFFH
F653 000000           NOP
F656 00               NOP
F657 D1               POP     D
F658 C9               RET


                ;       PHYSICAL DEVICE DRIVER ROUTINES

                ;       REQUIREMENTS
                ;               MAINTAIN CONTENTS OF ALL
                ;               REGISTERS EXCEPT A AND F.
                ;               EXIT BY RETURN INST.
```

```
                    ;       VIDEO DRIVER

F659 79             CRT:    MOV     A,C
F65A B7                     ORA     A               ;CHECK FOR NULL
F65B C8                     RZ
F65C E5                     PUSH    H
F65D CD36F6                 CALL    ADIOB
F66Ø 2B                     DCX     H
F661 2B                     DCX     H
F662 2B                     DCX     H
F663 C3Ø4F7                 JMP     ØF7Ø4H

                    ;       KEYBOARD DRIVER

F666 DBØ2           KYBD:   IN      2
F668 E6Ø1                   ANI     1
F66A C266F6                 JNZ     KYBD
F66D DBØ3                   IN      3
F66F E67F                   ANI     7FH
F671 FE61                   CPI     61H             ;LOWER CASE  A
F673 DA7DF6                 JC      KB1
F676 FE7B                   CPI     7AH+1           ;LOWER CASE Z +1
F678 D27DF6                 JNC     KB1
F67B E6DF                   ANI     ØDFH            ;DELET ONE BIT
F67D B7             KB1:    ORA     A
F67E C9                     RET

                    ;       KEYBOARD STATUS DRIVER

F67F DBØ2           KYST:   IN      2
F681 E6Ø1                   ANI     1
F683 D6Ø1                   SUI     1
F685 9F                     SBB     A
F686 C9                     RET

                    ;       READER DRIVER

F687 E5             RDR:    PUSH    H
F688 21ØØØØ                 LXI     H,Ø
F68B DBØ4           RD:     IN      4
F68D E6Ø1                   ANI     1
F68F CA9BF6                 JZ      RD2
F692 2B                     DCX     H
F693 7C                     MOV     A,H
F694 B5                     ORA     L
F695 C28BF6                 JNZ     RD
F698 37                     STC
F699 E1                     POP     H
F69A C9                     RET
```

```
F69B DBØ5       RD2:     IN        5
F69D B7                  ORA       A
F69E E1                  POP       H
F69F C9                  RET

                ;        TELETYPE STATUS DRIVER

F6AØ DBØ3       TTST:    IN        3
F6A2 E6Ø2                ANI       2
F6A4 D6Ø2                SUI       2
F6A6 9F                  SBB       A
F6A7 C9                  RET

                ;        TELETYPE INPUT DRIVER

F6A8 AF         TTI:     XRA       A
F6A9 D3ØØ                OUT       Ø
F6AB DBØ3       TTI1:    IN        3
F6AD E6Ø2                ANI       2
F6AF CAABF6              JZ        TTI1
F6B2 DBØ2                IN        2
F6B4 E67F                ANI       7FH
F6B6 C9                  RET

                ;        TELETYPE OUTPUT DRIVER

F6B7 DB03       TTO:     IN        3
F6B9 E601                ANI       1
F6BB CAB7F6              JZ        TTO
F6BE 79                  MOV       A,C
F6BF D3Ø2                OUT       2
F6C1 C9                  RET

                ;        TELETYPE READER DRIVER

F6C2 3EØ1       TTR:     MVI       A,1
F6C4 D3ØØ                OUT       Ø
F6C6 3EØØ                MVI       A,Ø
F6C8 D3ØØ                OUT       Ø
F6CA DBØØ       TTR1:    IN        Ø
F6CC E6Ø1                ANI       1
F6CE C2CAF6              JNZ       TTR1
F6D1 DBØ1                IN        1
F6D3 C9                  RET
```

```
                    ;         THERMAL PRINTER DRIVER

F6D4 DBØ2           THRM:     IN       2
F6D6 E68Ø                     ANI      8ØH
F6D8 C2D4F6                   JNZ      THRM
F6DB 79                       MOV      A,C
F6DC D3Ø3                     OUT      3
F6DE C9                       RET

                    ;         PUNCH DRIVER

F6DF DBØ4           PUNCH:    IN       4
F6E1 E68Ø                     ANI      8ØH
F6E3 C2DFF6                   JNZ      PUNCH
F6E6 79                       MOV      A,C
F6E7 D3Ø5                     OUT      5
F6E9 C9                       RET
```

CP/M ASSEMBLER - VER 1.0

```
F800                   ORG     0F800H
F800 2109F8     ENTRY: LXI     H,MESS
F803 CD1EF0            CALL    STRING
F806 C321F0            JMP     RENT
F01E =         STRING: EQU     0F01EH
                       ;
F809 0D0A202020MESS:   DB      0DH,0AH,'     HELLO !!   YOU HAVE ENTERED THE '
F830 0A0D574F52        DB      0AH,0DH,'WORLD OF DIAGNOSTICS.  THIS LIST WILL ACQUAINT'
F860 0D0A594F55        DB      0DH,0AH,'YOU WITH SOME OF THE COMMANDS OF THE DIAGNOSTIC'
F891 0D0A4F5045        DB      0DH,0AH,'OPERATING SYSTEM.  MANY OF THE FUNCTIONS ARE VERY'
F8C4 0D0A53494D        DB      0DH,0AH,'SIMILAR TO CPM/DDT.'
F8D9 0D0A434F4D        DB      0DH,0AH,'COMMAND                   FUNCTION'
F8FC 0D0A              DB      0DH,0AH
F8FE 0D0A202020        DB      0DH,0AH,'    A           ASSIGNS I/O DEVICES ( PHYSICAL'
F92E 0D0A202020        DB      0DH,0AH,'                   TO LOGICAL DEVICE )'
F956 0D0A              DB      0DH,0AH
F958 0D0A202020        DB      0DH,0AH,'    B           DUMP MEMORY IN BINARY ON PUNCH DEVICE'
F98F 0D0A              DB      0DH,0AH
F991 0D0A202020        DB      0DH,0AH,'    C           HEXADECIMAL ARITHMETIC'
F9B9 0D0A              DB      0DH,0AH
F9BB 0D0A202020        DB      0DH,0AH,'    D           DISPLAY A BLOCK OF MEMORY'
F9E6 0D0A              DB      0DH,0AH
F9E8 0D0A202020        DB      0DH,0AH,'    F           FILLS A BLOCK OF MEMORY WITH A CONSTANT'
FA21 0D0A              DB      0DH,0AH
FA23 0D0A202020        DB      0DH,0AH,'    G           GO TO ADDRESS AND EXECUTE, OPTIONAL'
FA58 0D0A202020        DB      0DH,0AH,'                   BREAK POINTS.'
FA7A 0A0D              DB      0AH,0DH
FA7C 0D0A202020        DB      0DH,0AH,'    H           HELP, THIS PROGRAM'
FAA0 0D0A              DB      0DH,0AH
FAA2 0D0A202020        DB      0DH,0AH,'    K           COPY FROM READER TO PUNCH'
FACD 0D0A              DB      0DH,0AH
FACF 0D0A202020        DB      0DH,0AH,'    L           LOAD BINARY TAPE, OPTIONAL BIAS'
FB00 0D0A              DB      0DH,0AH
FB02 0D0A202020        DB      0DH,0AH,'    M           MOVE A BLOCK OF MEMORY TO ANOTHER LOC'
FB39 0D0A              DB      0DH,0AH
FB3B 0D0A202020        DB      0DH,0AH,'    N           OUTPUTS 60 NULLS TO PUNCH DEVICE'
FB6D 0D0A              DB      0DH,0AH
FB6F 0D0A202020        DB      0DH,0AH,'    R           LOAD A HEX TAPE FROM READER DEVICE'
FBA3 0D0A              DB      0DH,0AH
FBA5 0D0A202020        DB      0DH,0AH,'    S           DISPLAY AND CHANGE ANY MEM LOC'
FBD5 0D0A              DB      0DH,0AH
FBD7 0D0A202020        DB      0DH,0AH,'    T           TEST LIST AND EXECUTION PROGRAM'
FC08 0D0A              DB      0DH,0AH
FC0A 0D0A202020        DB      0DH,0AH,'    W           DUMP MEMORY IN HEX ON PUNCH DEVICE'
FC3E 0D0A              DB      0DH,0AH
FC40 0D0A202020        DB      0DH,0AH,'    X           CPU REGISTER DISPLAY AND CHANGE',0
F021 =         RENT:   EQU     0F021H
               ;
               ;
               ;

FD00                   ORG     0FD00H
               ;
FD00 DBFF      ENTRY1: IN      0FFH
FD02 E601              ANI     01H
FD04 C213FD            JNZ     FART
FD07 219AFD    ENTRY2: LXI     H,MESS2
FD0A CD1EF0            CALL    STRING          ;PRINT LIST OF TESTS
FD0D 2153FD    ENTRY3: LXI     H,MESS3
FD10 CD1EF0            CALL    STRING          ;PROMT FOR INPUT
FD13 CD2FF1    FART:   CALL    TI
FD16 FE03              CPI     03H             ;TEST IF CONTROL C
FD18 CA21F0            JZ      RENT
FD1B FE40              CPI     40H
FD1D DA2EFD            JC      NUM             ;NUMBER
FD20 FE50              CPI     50H
FD22 DA3EFD            JC      LETTER
```

```
                        ;
FD25 214DFD     ERROR:  LXI     H,MESS4
FD28 CD1EF0             CALL    STRING          ;PRINT ERROR >?
FD2B C30DFD             JMP     ENTRY3          ;REPROMT
                        ;
                        ;
FD2E FE30       NUM:    CPI     30H             ;SEE IF NUMBER
FD30 DA25FD             JC      ERROR           ;JUMP IF NO
FD33 FE3A               CPI     3AH
FD35 E60F               ANI     0FH             ;REMOVE LEAD NIBBLE
FD37 87                 ADD     A               ;DOUBLE FOR TABLE LOOK UP
FD38 21DDFE             LXI     H,NUMTAB        ;NUMBER TABLE
FD3B C344FD             JMP     COMMON
                        ;
                        ;
FD3E E60F       LETTER: ANI     0FH
FD40 87                 ADD     A               ;DOUBLE FOR TABLE
FD41 21F1FE             LXI     H,LETTAB        ;LETTER TABLE
FD44 1600       COMMON: MVI     D,0
FD46 5F                 MOV     E,A
FD47 19                 DAD     D               ;ADD OFFSET TO TABLE ADDRESS
FD48 7E                 MOV     A,M
FD49 23                 INX     H
FD4A 66                 MOV     H,M
FD4B 6F                 MOV     L,A
FD4C E9                 PCHL                    ;JUMP TO TEST PROGRAM
                        ;
                        ;
FD4D 0D0A203F20MESS4:   DB      0DH,0AH,' ? ',0
FD53 0D0A454E54MESS3    DB      0DH,0AH,'ENTER TEST ID NO.  TO RUN TEST'
FD73 0D0A454E54         DB      0DH,0AH,'ENTER CONTROL C TO RETURN TO MONITOR',0
FD9A 0D0A0A2054MESS2:   DB      0DH,0AH,0AH,' TESTS AVAILABLE'
FDAD 0D0A202031         DB      0DH,0AH,'  1 -  COMPREHENSIVE MEMORY TEST'
FDCF 0D0A202032         DB      0DH,0AH,'  2 -  MINI-MEMORY  0 - .1K'
FDEC 0D0A202033         DB      0DH,0AH,'  3 -  MINI-MEMORY  0 -  8K'
FE09 0D0A202034         DB      0DH,0AH,'  4 -  MINI-MEMORY  0 - 24K'
FE26 0D0A202035         DB      0DH,0AH,'  5 -  FORMATTED DISK TEST'
FE42 0D0A202036         DB      0DH,0AH,'  6 -  DISK TRACK READ'
FE5A 0D0A202037         DB      0DH,0AH,'  7 -  DISK TRACK WRITE'
FE73 0D0A202038         DB      0DH,0AH,'  8 -  UNIBUS PORT TEST'
FE8C 0D0A202039         DB      0DH,0AH,'  9 -  UNIBUS COMMUNICATION TEST'
FEAE 0D0A202041         DB      0DH,0AH,'  A -  UNIBUS SNAPSHOT'
FEC6 0D0A202042         DB      0DH,0AH,'  B -  DISPLAY TESTS',0
                        ;
                        ;ADD MORE TO DIRECTORY HERE
                        ;
F12F =                  TI:     EQU     0F12FH
                        ;
FEDD 25FD       NUMTAB: DW      ERROR
FEDF 00C0               DW      0C000H          ;MEMORY TEST
FEE1 90C2               DW      0C290H          ;MINI .1K
FEE3 00D6               DW      0D600H          ;MINI   8K
FEE5 00D7               DW      0D700H          ;MINI   24K
FEE7 00C8               DW      0C800H          ;FOMAT DSK
FEE9 40CE               DW      0CE40H          ;TRK RD
FEEB 80CD               DW      0CD80H          ;TRK WRT
FEED 00D0               DW      0D000H          ;UB PORT
FEEF 00D1               DW      0D100H          ;UB COMM
FEF1 25FD       LETTAB: DW      ERROR
FEF3 00D5               DW      0D500H          ;SNAPSHOT
FEF5 00E0               DW      0E000H          ;DISPLAY
FEF7 25FD               DW      ERROR
FEF9 25FD               DW      ERROR
FEFB 25FD               DW      ERROR
FEFD 25FD               DW      ERROR
FEFF 25FD               DW      ERROR
FF01 25FD               DW      ERROR
FF03 25FD               DW      ERROR
FF05 25FD               DW      ERROR
FF07
```

END

DATE

FILMED

1-81

DTIC